```
SSSSSSSSSSSS   YYY         YYY   SSSSSSSSSSSS
SSSSSSSSSSSS   YYY         YYY   SSSSSSSSSSSS
SSSSSSSSSSSS   YYY         YYY   SSSSSSSSSSSS
SSS            YYY         YYY   SSS
SSS            YYY         YYY   SSS
SSS            YYY         YYY   SSS
SSS              YYY     YYY     SSS
SSS              YYY     YYY     SSS
SSS              YYY     YYY     SSS
   SSSSSSSS        YYY             SSSSSSSS
   SSSSSSSS        YYY             SSSSSSSS
   SSSSSSSS        YYY             SSSSSSSS
        SSS        YYY                  SSS
        SSS        YYY                  SSS
        SSS        YYY                  SSS
        SSS        YYY                  SSS
        SSS        YYY                  SSS
        SSS        YYY                  SSS
SSSSSSSSSSSS       YYY           SSSSSSSSSSSS
SSSSSSSSSSSS       YYY           SSSSSSSSSSSS
SSSSSSSSSSSS       YYY           SSSSSSSSSSSS
```

```
EEEEEEEEEE  XX       XX   CCCCCCC   EEEEEEEEEE  PPPPPPPP   TTTTTTTTTT   IIIIII     000000    NN      NN
EEEEEEEEEE  XX       XX   CCCCCCCC  EEEEEEEEEE  PPPPPPPP   TTTTTTTTTT   IIIIII     000000    NN      NN
EE          XX       XX   CC        EE          PP     PP      TT          II     00    00  NN      NN
EE          XX       XX   CC        EE          PP     PP      TT          II     00    00  NN      NN
EE            XX   XX     CC        EE          PP     PP      TT          II     00    00  NNNN    NN
EE            XX   XX     CC        EE          PP     PP      TT          II     00    00  NNNN    NN
EEEEEEEE        XX        CC        EEEEEEEE    PPPPPPPP       TT          II     00    00  NN  NN  NN
EEEEEEEE        XX        CC        EEEEEEEE    PPPPPPPP       TT          II     00    00  NN  NN  NN
EE            XX   XX     CC        EE          PP             TT          II     00    00  NN    NNNN
EE            XX   XX     CC        EE          PP             TT          II     00    00  NN    NNNN
EE          XX       XX   CC        EE          PP             TT          II     00    00  NN      NN
EE          XX       XX   CC        EE          PP             TT          II     00    00  NN      NN  ....
EEEEEEEEEE  XX       XX   CCCCCCCC  EEEEEEEEEE  PP             TT       IIIIII     000000    NN      NN  ....
EEEEEEEEEE  XX       XX   CCCCCCCC  EEEEEEEEEE  PP             TT       IIIIII     000000    NN      NN  ....


LL            IIIIII     SSSSSSSS
LL            IIIIII     SSSSSSSS
LL              II     SS
LL              II     SS
LL              II     SS
LL              II     SS
LL              II       SSSSSS
LL              II       SSSSSS
LL              II            SS
LL              II            SS
LL              II            SS
LL              II            SS
LLLLLLLLLL    IIIIII     SSSSSSSS
LLLLLLLLLL    IIIIII     SSSSSSSS
```

```
0000     1              .TITLE  EXCEPTION - EXCEPTION HANDLING
0000     2              .IDENT  'V04-000'
0000     3
0000     4
0000     5      ;*******************************************************************
0000     6      ;*                                                                 *
0000     7      ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                        *
0000     8      ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.         *
0000     9      ;*  ALL RIGHTS RESERVED.                                           *
0000    10      ;*                                                                 *
0000    11      ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *
0000    12      ;*  ONLY IN  ACCORDANCE  WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE *
0000    13      ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER *
0000    14      ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *
0000    15      ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY *
0000    16      ;*  TRANSFERRED.                                                   *
0000    17      ;*                                                                 *
0000    18      ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE *
0000    19      ;*  AND  SHOULD  NOT  BE  CONSTRUED  AS  A COMMITMENT BY DIGITAL EQUIPMENT *
0000    20      ;*  CORPORATION.                                                   *
0000    21      ;*                                                                 *
0000    22      ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS *
0000    23      ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.        *
0000    24      ;*                                                                 *
0000    25      ;*                                                                 *
0000    26      ;*******************************************************************
0000    27
0000    28      ; D. N. CUTLER 6-JUL-76
0000    29
0000    30      ; MODIFIED BY:
0000    31      ;
0000    32      ;     V03-007 WMC0002          Wayne Cardoza             28-Aug-1984
0000    33      ;             Mask spurious accvio caused by 780/785 hardware bug.
0000    34      ;
0000    35      ;     V03-006 LY00B2           Larry Yetto               10-FEB-1984 09:56
0000    36      ;             Fix truncation error
0000    37      ;
0000    38      ;     V03-005 LJK0260          Lawrence J. Kenah         5-Feb-1984
0000    39      ;             Allow exception dispatching to take a detour through an
0000    40      ;             instruction emulator so that the exception parameters can
0000    41      ;             be modified to describe an exception at the site of the
0000    42      ;             emulated instruction rather than inside the emulator.
0000    43      ;             Correct errors that appeared in comments.
0000    44      ;
0000    45      ;     V03-004 ACG0393          Andrew C. Goldstein,      20-Jan-1984  1:38
0000    46      ;             Fix FP validation in condition handler search; also
0000    47      ;             call EXE$UNWIND directly to avoid P1 vectors
0000    48      ;
0000    49      ;     V03-003 WMC0001          Wayne Cardoza             28-Oct-1983
0000    50      ;             Change mode to user or supervisor handlers should not get
0000    51      ;             control in privileged modes.
0000    52      ;
0000    53      ;     V03-002 ACG0348          Andrew C. Goldstein,      4-Aug-1983  17:01
0000    54      ;             Fix unwinding to frame of exception
0000    55      ;
0000    56      ;     V03-001 ACG0310          Andrew C. Goldstein,      31-Jan-1983  13:44
0000    57      ;             Fix probing of stack after expansion
```

```
                    0000      58 ;
                    0000      59 ;
                    0000      60 ; HARDWARE EXCEPTION HANDLING
                    0000      61 ;
                    0000      62 ;**************************************************************************
                    0000      63 ;
                    0000      64 ; FAIR WARNING!! THE EXCEPTION REFLECTION AND CONDITION HANDLING CODE IN
                    0000      65 ; THIS MODULE CRAWLS WITH ASSUMPTIONS ABOUT THE FORMAT OF THE STACK AND
                    0000      66 ; ARGUMENT LISTS, AS DOCUMENTED IN VARIOUS COMMENTS THROUGHOUT. SINCE
                    0000      67 ; THE STACK POINTER MOVES FREQUENTLY, NO ATTEMPT HAS BEEN MADE TO USE
                    0000      68 ; SYMBOLIC OFFSETS FOR STACK RELATIVE REFERENCES. CHANGES TO THE STACK
                    0000      69 ; FORMAT SHOULD BE MADE ONLY AFTER THOROUGH INSPECTION AND UNDERSTANDING
                    0000      70 ; OF THE CODE (NOT TO MENTION APPENDIX C OF THE ARCHITECTURE HANDBOOK).
                    0000      71 ; NOTE ALSO THAT LIB$SIGNAL MUST TRACK THE STACK FORMATS USED HERE.
                    0000      72 ;
                    0000      73 ;**************************************************************************
                    0000      74 ;
                    0000      75 ; MACRO LIBRARY CALLS
                    0000      76 ;
                    0000      77
                    0000      78         $CHFDEF                          ;DEFINE CONDITION HANDLING ARGLIST OFFSETS
                    0000      79         $IPLDEF                          ;DEFINE INTERRUPT PRIORITY LEVELS
                    0000      80         $MCHKDEF                         ;DEFINE MACHINE CHECK RECOVERY BITS
                    0000      81         $PCBDEF                          ;DEFINE PCB OFFSETS
                    0000      82         $PHDDEF                          ;DEFINE PHD OFFSETS
                    0000      83         $PRDEF                           ;DEFINE PROCESSOR REGISTERS
                    0000      84         $PSLDEF                          ;DEFINE PROCESSOR STATUS FIELDS
                    0000      85         $SSDEF                           ;DEFINE SYSTEM STATUS VALUES
                    0000      86         $STSDEF                          ;DEFINE STATUS CODE FIELDS
                    0000      87         $VADEF                           ;DEFINE VIRTUAL ADDRESS FIELDS
                    0000      88
                    0000      89 ;
                    0000      90 ; LOCAL SYMBOLS
                    0000      91 ;
                    0000      92 ; CALL FRAME OFFSET DEFINITIONS
                    0000      93 ;
                    0000      94
          00000000  0000      95 HANDLER=0                               ;CONDITION HANDLER ADDRESS
          00000004  0000      96 SAVPSW=4                                ;SAVED PSW FROM CALL
          00000006  0000      97 SAVMSK=6                                ;REGISTER SAVE MASK
          00000008  0000      98 SAVAP=8                                 ;SAVED AP REGISTER IMAGE
          0000000C  0000      99 SAVFP=12                                ;SAVED FP REGISTER IMAGE
          00000010  0000     100 SAVPC=16                                ;SAVED PC REGISTER IMAGE
          00000014  0000     101 SAVRG=20                                ;OTHER SAVED REGISTER IMAGES
                    0000     102
                    0000     103 ;
                    0000     104 ; LOCAL DATA
                    0000     105 ;
                    0000     106
          00000000  107              .PSECT  YEXEPAGED1,LONG
                    0000     108
          00000000  0000     109 FINAL_IDX       = 0                     ;INDICES TO FETCH MESSAGE ADDRESSES
          00000001  0000     110 ATTCONSTO_IDX   = 1
          00000002  0000     111 BADHANDLER_IDX  = 2
          00000003  0000     112 BADAST_IDX      = 3
                    0000     113
                    0000     114 MSG_VECTOR:
```

```
        00000010' 0000   115        .ADDRESS      FINALMSG
        00000033' 0004   116        .ADDRESS      ATTCONSTO_MSG
        00000052' 0008   117        .ADDRESS      BADHANDLER_MSG
        00000083' 000C   118        .ADDRESS      BADAST_MSG
                  0010   119
                  0010   120 FINALMSG:                                    ;FINAL EXCEPTION MESSAGE
72 6F 20 6B 63 61 74 73 20 64 61 62  0010   121        .ASCIZ  /bad stack or no handler specified./ ;
20 72 65 6C 64 6E 61 68 20 6F 6E 20  001C
   00 2E 64 65 69 66 69 63 65 70 73  0028
                  0033   122 ATTCONSTO_MSG:                               ;ATTEMPT TO CONTINUE FROM STOP MESSAGE
63 20 6F 74 20 74 70 6D 65 74 74 61  0033   123        .ASCIZ  /attempt to continue from STOP./
6D 6F 72 66 20 65 75 6E 69 74 6E 6F  003F
         00 2E 50 4F 54 53 20        004B
                  0052   124 BADHANDLER_MSG:                              ;BAD CONDITION HANDLER MESSAGE ADDRESS
64 6E 6F 63 20 64 69 6C 61 76 6E 69  0052   125        .ASCIZ  /invalid condition handler address or entry mask./
65 6C 64 6E 61 68 20 6E 6F 69 74 69  005E
72 6F 20 73 73 65 72 64 64 61 20 72  006A
2E 6B 73 61 6D 20 79 72 74 6E 65 20  0076
                     00  0082
                  0083   126 BADAST_MSG:                                  ;BAD AST MESSAGE ADDRESS
20 54 53 41 20 64 69 6C 61 76 6E 69  0083   127        .ASCIZ  /invalid AST address or entry mask./
65 20 72 6F 20 73 73 65 72 64 64 61  008F
   00 2E 6B 73 61 6D 20 79 72 74 6E  009B
```

EXCEPTION
V04-000

C 7

- EXCEPTION HANDLING
ACCESS VIOLATION FAULT

16-SEP-1984 00:06:28   VAX/VMS Macro V04-00   Page   4
5-SEP-1984 03:41:43   [SYS.SRC]EXCEPTION.MAR;1   (2)

```
                        00A6    129            .SBTTL  ACCESS VIOLATION FAULT
                        00A6    130    ;+
                        00A6    131    ; EXE$ACVIOLAT - ACCESS VIOLATE FAULT
                        00A6    132    ;
                        00A6    133    ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN AN ACCESS VIOLATION IS
                        00A6    134    ; DETECTED. THE STATE OF THE STACK ON ENTRY IS:
                        00A6    135    ;
                        00A6    136    ;       00(SP) = ACCESS VIOLATION REASON MASK.
                        00A6    137    ;       04(SP) = ACCESS VIOLATION VIRTUAL ADDRESS.
                        00A6    138    ;       08(SP) = EXCEPTION PC.
                        00A6    139    ;       12(SP) = EXCEPTION PSL.
                        00A6    140    ;
                        00A6    141    ; ACCESS VIOLATION REASON MASK FORMAT IS:
                        00A6    142    ;
                        00A6    143    ;       BIT 0 = TYPE OF ACCESS VIOLATION.
                        00A6    144    ;               0 = PTE ACCESS CODE DID NOT PERMIT INTENTED ACCESS.
                        00A6    145    ;               1 = P0LR, P1LR, OR S0LR LENGTH VIOLATION.
                        00A6    146    ;       BIT 1 = PTE REFERENCE.
                        00A6    147    ;               0 = SPECIFIED VIRTUAL ADDRESS NOT ACCESSIBLE.
                        00A6    148    ;               1 = ASSOCIATED PAGE TABLE ENTRY NOT ACCESSIBLE.
                        00A6    149    ;       BIT 2 = INTENDED ACCESS TYPE.
                        00A6    150    ;               0 = READ.
                        00A6    151    ;               1 = MODIFY.
                        00A6    152    ;
                        00A6    153    ; THE EXCEPTION NAME FOLLOWED BY THE NUMBER OF EXCEPTION ARGUMENTS ARE
                        00A6    154    ; PUSHED ON THE STACK. FINAL PROCESSING IS ACCOMPLISHED IN COMMON CODE.
                        00A6    155    ;-
                        00A6    156
                     00000000    157            .PSECT  $AEXENONPAGED,LONG
                        0000    158            .ALIGN  LONG
                        0000    159    EXE$ACVIOLAT::                             ;ACCESS VIOLATION FAULTS
03   0C AE   02   18 ED 0000    160            CMPZV   #PSL$V_CURMOD,#PSL$S_CURMOD,12(SP),#PSL$C_USER ; CHECK FOR USER
                  17 12 0006    161            BNEQ    20$                        ;NO, REALLY AN ACCESS VIOLATION
                  0F BB 0008    162            PUSHR   #^M<R0,R1,R2,R3>           ;SAVE WORKING REGISTERS
         52   14 AE D0 000A    163            MOVL    <4+<4*4>>(SP),R2           ;GET BASE ADDRESS FOR EXTEND
     0000041A'EF   16 000E    164            JSB     EXE$EXPANDSTK              ;EXPAND STACK
            06 50 E9 0014    165            BLBC    R0,10$                     ;BR IF CANT EXTEND
               0F BA 0017    166            POPR    #^M<R0,R1,R2,R3>           ;RESTORE REGISTERS
            5E 08 C0 0019    167    5$:     ADDL    #8,SP                      ;CLEAN EXCEPTION PARAMETERS FROM STACK
                  02 001C    168            REI                                ;AND RETURN TO RETRY INSTRUCTION
               0F BA 001D    169    10$:    POPR    #^M<R0,R1,R2,R3>           ;RESTORE REGISTERS
                     001F    170    20$:
                     001F    171    ;+
                     001F    172    ; THE FOLLOWING SECTION OF CODE IS USED TO MASK A SPURIOUS ACCESS VIOLATION
                     001F    173    ; WHICH IS SEEN ON THE 780/785.  AT THIS TIME THE EXACT CAUSE IS UNKNOWN.  IT IS
                     001F    174    ; ASSUMED TO BE EITHER HARDWARE OR MICROCODE.
                     001F    175    ;
                     001F    176    ; THE SPURIOUS ACCVIO ONLY OCCURS WHEN AN INSTRUCTION STARTS ON THE LAST BYTE
                     001F    177    ; OF A PAGE.  THE EXACT CIRCUMSTANCES ARE UNKNOWN SINCE IT CANNOT BE RELIABLY
                     001F    178    ; REPRODUCED.
                     001F    179    ;-
04 AE   01FF 8F B3 001F    180            BITW    #^X1FF,4(SP)               ; WAS FIRST BYTE OF PAGE REFERENCED
               3C 12 0025    181            BNEQ    ACVIOLAT                   ; NO
               6E D5 0027    182            TSTL    (SP)                       ; WAS IT READ ACCESS
               38 12 0029    183            BNEQ    ACVIOLAT                   ; NO
7E   04 AE   08 AE C3 002B    184            SUBL3   8(SP),4(SP),-(SP)          ; GET DIFFERENCE BETWEEN PC, REFERENCE
            01 8E D1 0031    185            CMPL    (SP)+,#1                   ; ARE THEY ADJACENT
```

```
                    2D     12  0034  186          BNEQ    ACVIOLAT                        ; NO
                           01  0036  187          NOP                                     ; LEAVE ROOM FOR AN INVALID IF WE NEED IT
                           01  0037  188          NOP
                           01  0038  189          NOP
                           01  0039  190          NOP
                           01  003A  191          NOP
                           01  003B  192          NOP
            04 BE   01  00 0C  003C  193          PROBER  #0,#1,@4(SP)                    ; IS IT SPURIOUS
                           20  13  0041  194       BEQL    ACVIOLAT                        ; NO
00000000'EF  00000000'EF  D1  0043  195          CMPL    EXE$GL_ABSTIM,EXE$GL_BADACV_T ; ARE THEY COMING FAST
                           13  13  004E  196       BEQL    ACVIOLAT                        ; YES
00000000'EF  00000000'EF  D0  0050  197          MOVL    EXE$GL_ABSTIM,EXE$GL_BADACV_T ; SAVE TIME OF THIS ONE
            00000000'EF  D6  005B  198          INCL    EXE$GL_BADACV_C                 ; COUNT THEM
                           B6  11  0061  199       BRB     5$                              ; GO RETRY
                               0063  200 ACVIOLAT:
         7E   0C   3C  0063  201          MOVZWL  #SS$_ACCVIO,-(SP)               ;SET EXCEPTION NAME
              00EE  31  0066  202          BRW     EX5ARG                          ;
```

```
                    0069   204           .SBTTL   ARITHMETIC TRAPS
                    0069   205  ;+
                    0069   206  ; EXE$ARITH - ARITHMETIC TRAPS
                    0069   207  ;
                    0069   208  ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN AN ARITHMETIC TRAP IS
                    0069   209  ; DETECTED AT THE END OF AN INSTRUCTION. THE STATE OF THE STACK ON ENTRY
                    0069   210  ; IS:
                    0069   211  ;
                    0069   212  ;         00(SP) = ARITHMETIC EXCEPTION CODE.
                    0069   213  ;         04(SP) = EXCEPTION PC.
                    0069   214  ;         08(SP) = EXCEPTION PSL.
                    0069   215  ;
                    0069   216  ; POSSIBLE ARITHMETIC EXCEPTION CODES ARE:
                    0069   217  ;
                    0069   218  ;         0 = UNDEFINED.
                    0069   219  ;         1 = INTERGER OVERFLOW.
                    0069   220  ;         2 = INTEGER DIVIDE BY ZERO.
                    0069   221  ;         3 = FLOATING OVERFLOW.
                    0069   222  ;         4 = FLOATING OR DECIMAL DIVIDE BY ZERO.
                    0069   223  ;         5 = FLOATING UNDERFLOW.
                    0069   224  ;         6 = DECIMAL STRING OVERFLOW.
                    0069   225  ;         7 = SUBSCRIPT RANGE TRAP IN INDEX INSTRUCTION.
                    0069   226  ;         8 = FLOATING OVERFLOW FAULT.
                    0069   227  ;         9 = FLOATING DIVIDE BY ZERO FAULT.
                    0069   228  ;        10 = FLOATING UNDERFLOW FAULT.
                    0069   229  ;
                    0069   230  ; THE EXCEPTION NAME FOLLOWED BY THE NUMBER OF EXCEPTION ARGUMENTS ARE
                    0069   231  ; PUSHED ON THE STACK. FINAL PROCESSING IS ACCOMPLISHED IN COMMON CODE.
                    0069   232  ;-
                    0069   233
                    0069   234           .ALIGN   LONG
                    006C   235  EXE$ARITH::                                   ;ARITHMETIC TRAPS
      6E   08   C4  006C   236           MULL    #8,(SP)                      ;CALCULATE EXCEPTION NAME OFFSET
 6E  0474 8F   A0  006F   237           ADDW    #S$$_ARTRES,(SP)             ;CALCULATE ACTUAL EXCEPTION NAME
          0104  31  0074   238           BRW     EX3ARG                       ;
```

EXCEPTION   - EXCEPTION HANDLING   16-SEP-1984 00:06:28   VAX/VMS Macro V04-00   Page 7
V04-000   AST DELIVERY STACK FAULT   5-SEP-1984 03:41:43   [SYS.SRC]EXCEPTION.MAR;1   (4)

                                                          F 7

```
                        0077  240              .SBTTL   AST DELIVERY STACK FAULT
                        0077  241  ;+
                        0077  242  ;  EXE$ASTFLT - AST DELIVERY STACK FAULT
                        0077  243  ;
                        0077  244  ;  THIS ROUTINE IS ENTERED VIA A JMP FROM THE AST DELIVERY MODULE
                        0077  245  ;  WHEN AN INVALID STACK IS DETECTED WHILE ATTEMPTING TO DELIVER AN
                        0077  246  ;  AST.   THE STATE OF THE STACK ON ENTRY IS:
                        0077  247  ;
                        0077  248  ;          00(SP) = SP VALUE AT FAULT (UNMODIFIED BY AST DELIVERY).
                        0077  249  ;          04(SP) = AST PARAMETER OF FAILED AST.
                        0077  250  ;          08(SP) = PC AT AST DELIVERY INTERRUPT.
                        0077  251  ;          12(SP) = PSL AT AST DELIVERY INTERRUPT.
                        0077  252  ;          16(SP) = PC TO WHICH AST WOULD HAVE BEEN DELIVERED.
                        0077  253  ;          20(SP) = PSL AT WHICH AST WOULD HAVE BEEN DELIVERED.
                        0077  254  ;
                        0077  255  ;  THE EXCEPTION NAME FOLLOWED BY THE NUMBER OF ARGUMENTS ARE PUSHED
                        0077  256  ;  ON THE STACK.  FINAL PROCESSING IS ACCOMPLISHED IN COMMON CODE.
                        0077  257  ;
                        0077  258  ;  NOTE THAT THE PREVIOUS MODE FIELD IN THE PSL IS KERNEL MODE IN THIS
                        0077  259  ;  ENTRY POINT, SINCE AST'S (AND AST FAULTS) RESULT FROM INTERRUPTS, AS
                        0077  260  ;  OPPOSED TO ALL OTHER EXCEPTIONS WHICH RESULT FROM FAULTS OR TRAPS.
                        0077  261  ;  SINCE THE EXCEPTION MAIN BODY EXPECTS PREVIOUS MODE TO BE THE MODE
                        0077  262  ;  IN WHICH THE EXCEPTION OCCURRED, AND THE MODE TO WHICH IT WILL BE
                        0077  263  ;  REPORTED, WE MUST FIX IT UP ACCORDINGLY.
                        0077  264  ;-
                        0077  265
                        0077  266  EXE$ASTFLT::                                      ;AST DELIVERY STACK FAULT
       7E   040C 8F  3C 0077  267              MOVZWL  #SS$_ASTFLT,-(SP)            ;PUSH EXCEPTION CODE
                  07  DD 007C 268              PUSHL   #7                           ;AND COUNT OF ARGUMENTS
7E  1C AE  FF3FFFFF 8F  CB 007E 269              BICL3   #^C<PSL$M_PRVMOD>,28(SP),-(SP) ;CONSTRUCT PROPER PSL
           018F'CF  9F 0087 270              PUSHAB  W^EXE$EXCEPTION              ;AND EXCEPTION ENTRY POINT
                  02 008B 271              REI                                  ;SET PSL AND ENTER COMMON EXCEPTION CODE
```

```
                       008C    273              .SBTTL  BREAKPOINT FAULT
                       008C    274   ;+
                       008C    275   ; EXE$BREAK - BREAKPOINT FAULT
                       008C    276   ;
                       008C    277   ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN A BREAKPOINT
                       008C    278   ; INSTRUCTION IS EXECUTED. THE STATE OF THE STACK ON ENTRY IS:
                       008C    279   ;
                       008C    280   ;      00(SP) = EXCEPTION PC.
                       008C    281   ;      04(SP) = EXCEPTION PSL.
                       008C    282   ;
                       008C    283   ; THE EXCEPTION NAME FOLLOWED BY THE NUMBER OF EXCEPTION ARGUMENTS ARE
                       008C    284   ; PUSHED ON THE STACK. FINAL PROCESSING IS ACCOMPLISHED IN COMMON CODE.
                       008C    285   ;-
                       008C    286
                       008C    287              .ALIGN  LONG
                       008C    288   EXE$BREAK::                             ;BREAKPOINT FAULT
  7E  0414 8F  3C      008C    289              MOVZWL  #SS$_BREAK,-(SP)     ;SET EXCEPTION NAME
            00E7  31   0091    290              BRW     EX3ARG               ;
```

```
                              0094    292                .SBTTL  CHANGE MODE TO SUPERVISOR TRAP
                              0094    293        ;+
                              0094    294        ; EXE$CMODSUPR - CHANGE MODE TO SUPERVISOR TRAP
                              0094    295        ;
                              0094    296        ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN A CHANGE MODE TO SUPER-
                              0094    297        ; VISOR INSTRUCTION IS EXECUTED. THE STATE OF THE STACK ON ENTRY IS:
                              0094    298        ;
                              0094    299        ;       00(SP) = CHANGE MODE CODE.
                              0094    300        ;       04(SP) = EXCEPTION PC.
                              0094    301        ;       08(SP) = EXCEPTION PSL.
                              0094    302        ;
                              0094    303        ; IF THE PROCESS HAS DECLARED A CHANGE MODE TO SUPERVISOR HANDLER, THEN THE
                              0094    304        ; EXCEPTION IS DIRECTLY VECTORED TO THE SPECIFIED HANDLER. ELSE THE EXCEPTION
                              0094    305        ; NAME FOLLOWED BY THE NUMBER OF EXCEPTION ARGUMENTS ARE PUSHED ON THE STACK.
                              0094    306        ; FINAL PROCESSING IS ACCOMPLISHED IN COMMON CODE.
                              0094    307        ;-
                              0094    308
                              0094    309                .ALIGN  LONG
                              0094    310        EXE$CMODSUPR::                                  ;CHANGE MODE TO SUPERVISOR TRAP
      7E    041C 8F    3C     0094    311                MOVZWL  #SS$_CMODSUPR,-(SP)             ;SET EXCEPTION NAME
            02    18   ED     0099    312                CMPZV   #PSL$V_CURMOD,#PSL$S_CURMOD,-
      02    0C AE           009C    313                        12(SP),#PSL$C_SUPER             ;WERE WE IN USER OR SUPER MODE
            08    19   009F    314                BLSS    20$                             ;NO - DON'T GIVE CONTROL TO HANDLER
  00000000'9F    DD   00A1    315                PUSHL   @#CTL$GL_CMSUPR                 ;GET CONTENTS OF CHANGE MODE VECTOR
            16    11   00A7    316                BRB     EXCCMD                          ;
            00DB  31   00A9    317    20$:        BRW     EXSSXT
```

I 7

```
                              00AC     319            .SBTTL  CHANGE MODE TO USER TRAP
                              00AC     320    ;+
                              00AC     321    ; EXE$CMODUSER - CHANGE MODE TO USER TRAP
                              00AC     322    ;
                              00AC     323    ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN A CHANGE MODE TO USER
                              00AC     324    ; INSTRUCTION IS EXECUTED. THE STATE OF THE STACK ON ENTRY IS:
                              00AC     325    ;
                              00AC     326    ;        00(SP) = CHANGE MODE CODE.
                              00AC     327    ;        04(SP) = EXCEPTION PC.
                              00AC     328    ;        08(SP) = EXCEPTION PSL.
                              00AC     329    ;
                              00AC     330    ; IF THE PROCESS HAS DECLARED A CHANGE MODE TO USER HANLDER, THEN THE EXCEPTION
                              00AC     331    ; IS VECTORED DIRECTLY TO THE SPECIFIED HANDLER. ELSE THE EXCEPTION NAME FOLLOWED
                              00AC     332    ; BY THE NUMBER OF ARGUMENTS ARE PUSHED ON THE STACK. FINAL PROCESSING IS ACCOM-
                              00AC     333    ; PLISHED IN COMMON CODE.
                              00AC     334    ; IF THIS CODE IS ENTERED IN OTHER THAN USER MODE, IT IS TREATED AS THOUGH NO
                              00AC     335    ; USER HANDLER WERE DELARED.
                              00AC     336    ;-
                              00AC     337
                              00AC     338            .ALIGN  LONG
                              00AC     339            .ENABL  LSB
                              00AC     340    EXE$CMODUSER::                                  ;CHANGE MODE TO USER TRAP
  7E    0424 8F    3C        00AC     341            MOVZWL  #SS$_CMODUSER,-(SP)             ;SET EXCEPTION NAME
        02    18    ED        00B1     342            CMPZV   #PSL$V_CURMOD,#PSL$S_CURMOD,-    ;WERE WE IN USER MODE
  03    0C AE                  00B4     343                    12(SP),#PSL$C_USER
              0F    12        00B7     344            BNEQ    20$                             ;NO - DON'T GIVE CONTROL TO HANDLER
  00000000'9F    DD          00B9     345            PUSHL   @#CTL$GL_CMUSER                  ;GET CONTENTS OF CHANGE MODE VECTOR
        04    13              00BF     346    EXCCMD: BEQL    10$                             ;IF EQL NO DISPATCHER SPECIFIED
        6E 8ED0              00C1     347            POPL    (SP)                            ;REMOVE EXCEPTION NAME FROM STACK
              05              00C4     348            RSB
  5E    04    C0              00C5     349    10$:    ADDL    #4,SP                           ;CLEAN STACK
        00BC    31            00C8     350    20$:    BRW     EXE$SXT                         ;
                              00CB     351            .DSABL  LSB
```

```
                                        00CB  353                   .SBTTL  COMPATIBILITY MODE FAULTS
                                        00CB  354          ;+
                                        00CB  355          ; EXE$COMPAT - COMPATIBILITY MODE FAULT
                                        00CB  356          ;
                                        00CB  357          ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN A COMPATIBILITY MODE
                                        00CB  358          ; EXCEPTION IS DETECTED. THE STATE OF THE STACK ON ENTRY IS:
                                        00CB  359          ;
                                        00CB  360          ;       00(SP) = COMPATIBILITY EXCEPTION CODE.
                                        00CB  361          ;       04(SP) = EXCEPTION PC.
                                        00CB  362          ;       08(SP) = EXCEPTION PSL.
                                        00CB  363          ;
                                        00CB  364          ; POSSIBLE COMPATIBILITY EXCEPTION CODES ARE:
                                        00CB  365          ;
                                        00CB  366          ;       0 = RESERVED INSTRUCTION EXECUTION.
                                        00CB  367          ;       1 = BPT INSTRUCTION EXECUTION.
                                        00CB  368          ;       2 = IOT INSTRUCTION EXECUTION.
                                        00CB  369          ;       3 = EMT INSTRUCTION EXECUTION.
                                        00CB  370          ;       4 = TRAP INSTRUCTION EXECUTION.
                                        00CB  371          ;       5 = ILLEGAL INSTRUCTION EXECUTION.
                                        00CB  372          ;       6 = ODD ADDRESS FAULT.
                                        00CB  373          ;       7 = TBIT TRAP.
                                        00CB  374          ;
                                        00CB  375          ; THE EXCEPTION NAME FOLLOWED BY THE NUMBER OF EXCEPTION ARGUMENTS ARE
                                        00CB  376          ; PUSHED ON THE STACK. FINAL PROCESSING IS ACCOMPLISHED IN COMMON CODE.
                                        00CB  377          ;-
                                        00CB  378
                                        00CB  379                   .ALIGN  LONG
                                        00CC  380          EXE$COMPAT::                                    ;COMPATIBILITY MODE FAULTS
    00000000'9F   50   7D               00CC  381                   MOVQ    R0,@#CTL$AL_CMCNTX      ;SAVE R0,R1 IN COMPATIBILITY CONTEXT REGION
50  00000008'9F   DE               00D3  382                   MOVAL   @#CTL$AL_CMCNTX+8,R0    ;GET ADDRESS OF COMPATIBILITY CONTEXT AREA
         80   52   7D               00DA  383                   MOVQ    R2,(R0)+               ;SAVE R2,R3
         80   54   7D               00DD  384                   MOVQ    R4,(R0)+               ;SAVE R4,R5
         80   56   D0               00E0  385                   MOVL    R6,(R0)+               ;SAVE R6
         80   8E   D0               00E3  386                   MOVL    (SP)+,(R0)+            ;SAVE EXCEPTION CODE AND REMOVE FROM STACK
         60   8E   7D               00E6  387                   MOVQ    (SP)+,(R0)            ;SAVE PC AND PSL AND REMOVE FROM STACK
03C00000 8F   DD               00E9  388                   PUSHL   #<PSL$C_USER@PSL$V_PRVMOD>!<PSL$C_USER@PSL$V_CURMOD> ;FABRICATE PSL
    00000000'9F   DD               00EF  389                   PUSHL   @#CTL$GL_CMHANDLR      ;COMPATIBILITY MODE HANDLER ADDRESS
              01   13               00F5  390                   BEQL    20$                    ;BRANCH IF NONE SPECIFIED
              02               00F7  391                   REI                            ;JUMP TO COMPATIBILITY HANDLER
                                        00F8  392
                                        00F8  393          ;
                                        00F8  394          ; NO COMPATIBILITY MODE HANDLER WAS DECLARED. RESTORE THE STACK AND
                                        00F8  395          ; SAVED REGISTER, AND CONTINUE THROUGH NORMAL EXCEPTION CODE.  R0 NOW
                                        00F8  396          ; POINTS TO THE SAVED PC IN THE COMPATIBILITY CONTEXT AREA.
                                        00F8  397          ;
                                        00F8  398
     6E   60   7D               00F8  399          20$:     MOVQ    (R0),(SP)             ;RESTORE EXCEPTION PC AND PSL
          70   DD               00FB  400                   PUSHL   -(R0)                  ;PUSH EXCEPTION CODE AGAIN
50   E4 A0   D0               00FD  401                   MOVL    -28(R0),R0            ;RESTORE R0 FROM TOP OF CONTEXT AREA
7E  042C 8F   3C               0101  402                   MOVZWL  #SS$_COMPAT,-(SP)     ;SET EXCEPTION NAME
          04   DD               0106  403          EX4ARG:  PUSHL   #4                     ;SET NUMBER OF SIGNAL ARGUMENTS
          0084   31               0108  404                   BRW     EXE$EXCEPTION         ;FINISH IN COMMON CODE
```

```
010B    406            .SBTTL   KERNEL STACK NOT VALID FAULT
010B    407    ;+
010B    408    ; EXE$KERSTKNV - KERNEL STACK NOT VALID FAULT
010B    409    ;
010B    410    ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN A KERNEL STACK NOT
010B    411    ; VALID IS DETECTED DURING A CHANGE MODE INSTRUCTION, DURING AN REI
010B    412    ; INSTRUCTION, OR DURING AN ATTEMPT TO PUSH EXCEPTION INFORMATION ON
010B    413    ; THE KERNEL STACK. THIS EXCEPTION RUNS ON THE INTERRUPT STACK. THE
010B    414    ; STATE OF THE STACK ON ENTRY IS:
010B    415    ;
010B    416    ;        00(SP) = EXCEPTION PC.
010B    417    ;        04(SP) = EXCEPTION PSL.
010B    418    ;
010B    419    ; A FATAL KERNEL STACK NOT VALID BUGCHECK IS DECLARED.
010B    420    ;-
010B    421
010B    422            .ALIGN  LONG
010C    423    EXE$KERSTKNV::                          ;KERNEL STACK NOT VALID FAULT
010C    424            BUG_CHECK KRNLSTAKNV,FATAL       ;KERNEL STACK NOT VALID
```

```
                        0110   426              .SBTTL  SIGNAL UNRECOVERABLE MACHINE CHECK
                        0110   427     ;+
                        0110   428     ; EXE$MCHECK - SIGNAL UNRECOVERABLE MACHINE CHECK
                        0110   429     ;
                        0110   430     ; THIS ROUTINE IS ENTERED FROM THE MACHINE CHECK HANDLER TO SIGNAL
                        0110   431     ; THE ERROR BACK TO EITHER USER OR SUPERVISOR MODE.  IF THE MACHINE
                        0110   432     ; CHECK OCCURRED IN EXEC OR KERNEL MODE CONTROL DOES NOT COME HERE
                        0110   433     ; AND INSTEAD THE MACHINE CHECK HANDLER BUGCHECKS.
                        0110   434     ; STATE OF THE STACK ON ENTRY IS:
                        0110   435     ;
                        0110   436     ;       00(SP) = EXCEPTION PC
                        0110   437     ;       04(SP) = EXCEPTION PSL
                        0110   438     ;
                        0110   439     ; POSSIBLE CODES ARE:
                        0110   440     ;
                        0110   441     ;
                        0110   442     ; NO ALIGNMENT IS NECESSARY HERE, REI GETS US HERE
                        0110   443     ;-
                        0110   444
                        0110   445     EXE$MCHECK::
7E  02BC 8F  3C  0110   446              MOVZWL  #SS$_MCHECK,-(SP)        ;SET EXCEPTION NAME
             64  11  0115   447              BRB     EX3ARG                  ;FINISH IN COMMON CODE
```

```
                           0117   449              .SBTTL  OPCODE RESERVED TO CUSTOMER FAULT
                           0117   450   ;+
                           0117   451   ; EXES$OPCCUS - OPCODE RESERVED TO CUSTOMER FAULT
                           0117   452   ;
                           0117   453   ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN AN OPCODE THAT IS RESERVED
                           0117   454   ; TO CUSTOMERS IS EXECUTED. THE STATE OF THE STACK ON ENTRY IS:
                           0117   455   ;
                           0117   456   ;        00(SP) = EXCEPTION PC.
                           0117   457   ;        04(SP) = EXCEPTION PSL.
                           0117   458   ;
                           0117   459   ; THE EXCEPTION NAME FOLLOWED BY THE NUMBER OF EXCEPTION ARGUMENTS ARE
                           0117   460   ; PUSHED ON THE STACK. FINAL PROCESSING IS ACCOMPLISHED IN COMMON CODE.
                           0117   461   ;-
                           0117   462
                           0117   463              .ALIGN  LONG
                           0118   464   EXES$OPCCUS::                                 ;OPCODE RESERVED TO CUSTOMERS FAULT
      7E   0434 8F   3C    0118   465              MOVZWL  #SS$_OPCCUS,-(SP)         ;SET EXCEPTION NAME
           5C   11   011D  466              BRB     EX3ARG                           ;
```

N 7

EXCEPTION   - EXCEPTION HANDLING   16-SEP-1984 00:06:28 VAX/VMS Macro V04-00   Page 15
V04-000   OPCODE RESERVED TO DIGITAL FAULT   5-SEP-1984 03:41:43 [SYS.SRC]EXCEPTION.MAR;1   (12)

```
                        011F    468                 .SBTTL  OPCODE RESERVED TO DIGITAL FAULT
                        011F    469 ;+
                        011F    470 ; EXE$OPCDEC - OPCODE RESERVED TO DIGITAL FAULT
                        011F    471 ;
                        011F    472 ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN AN OPCODE THAT IS RESERVED
                        011F    473 ; TO DIGITAL IS EXECUTED. THE STATE OF THE STACK ON ENTRY IS:
                        011F    474 ;
                        011F    475 ;       00(SP) = EXECPTION PC.
                        011F    476 ;       04(SP) = EXCEPTION PSL.
                        011F    477 ;
                        011F    478 ; THE EXCEPTION NAME FOLLOWED BY THE NUMBER OF EXCEPTION ARGUMENTS ARE
                        011F    479 ; PUSHED ON THE STACK. FINAL PROCESSING IS ACCOMPLISHED IN COMMON CODE.
                        011F    480 ;-
                        011F    481
                        011F    482                 .ALIGN  LONG
                        0120    483 EXE$OPCDEC::                                     ;OPCODE RESERVED TO DIGITAL FAULT
  00 BE   FF 8F   91    0120    484         CMPB    #^XFF,a(SP)                      ;POSSIBLY A BUG CHECK?
           07    13    0125    485         BEQL    20$                              ;IF EQL YES
  7E   043C 8F   3C    0127    486 10$:    MOVZWL  #SS$_OPCDEC,-(SP)                ;SET EXCEPTION NAME
           4D    11    012C    487         BRB     EX3ARG
           6E    DD    012E    488 20$:    PUSHL   (SP)                             ;COPY ADDRESS OF INSTRUCTION
           6E    D6    0130    489         INCL    (SP)                             ;CALCULATE ADDRESS OF NEXT BYTE
                 0132    490         IFNORD  #1,a(SP),40$                     ;CAN NEXT BYTE BE READ?
  00 BE   FE 8F   91    0139    491         CMPB    #^XFE,a(SP)                      ;BUGCHECK WORD?
           08    13    013E    492         BEQL    30$                              ;IF EQL YES
  9E   FD 8F   91    0140    493         CMPB    #^XFD,a(SP)+                     ;BUGCHECK LONG?
           E1    12    0144    494         BNEQ    10$                              ;IF NEG NO
           7E    D5    0146    495         TSTL    -(SP)                            ;ADJUST STACK POINTER
           8E    D5    0148    496 30$:    TSTL    (SP)+                            ;REMOVE INSTRUCTION ADDRESS FROM STACK
         FEB3'    31    014A    497         BRW     EXE$BUG_CHECK
           00    DD    014D    498 40$:    PUSHL   #0                               ;SET REASON FOR ACCESS VIOLATION
         FEAE    31    014F    499         BRW     EXE$ACVIOLAT                     ;
```

```
                         0152    501                    .SBTTL  PAGE READ FAULT
                         0152    502          ;+
                         0152    503          ; EXE$PAGRDERR - PAGE READ FAULT
                         0152    504          ;
                         0152    505          ; THIS ROUTINE IS ENTERED VIA A JUMP FROM THE TRANSLATION NOT VALID
                         0152    506          ; EXCEPTION ROUTINE WHEN A READ ERROR OCCURS IN TRYING TO MAKE THE
                         0152    507          ; DESIRED PAGE VALID.  THE STATE OF THE STACK ON ENTRY IS:
                         0152    508          ;
                         0152    509          ;        00(SP) = TRANSLATION NOT VALID REASON MASK.
                         0152    510          ;        04(SP) = TRANSLATION NOT VALID VIRTUAL ADDRESS.
                         0152    511          ;        08(SP) = EXCEPTION PC.
                         0152    512          ;        12(SP) = EXCEPTION PSL.
                         0152    513          ;
                         0152    514          ; TRANSLATION NOT VALID REASON MASK FORMAT IS:
                         0152    515          ;
                         0152    516          ;        BIT 0 = 0 (USED ONLY FOR ACCESS VIOLATION).
                         0152    517          ;        BIT 1 = PTE REFERENCE.
                         0152    518          ;                0 = SPECIFIED VIRTUAL ADDRESS NOT VALID.
                         0152    519          ;                1 = ASSOCIATED PAGE TABLE ENTRY NOT VALID.
                         0152    520          ;        BIT 2 = INTENDED ACCESS TYPE.
                         0152    521          ;                0 = READ.
                         0152    522          ;                1 = MODIFY.
                         0152    523          ;
                         0152    524          ; THE EXCEPTION NAME FOLLOWED BY THE NUMBER OF EXCEPTION ARGUMENTS ARE
                         0152    525          ; PUSHED ON THE STACK. FINAL PROCESSING IS ACCOMPLISHED IN COMMON CODE.
                         0152    526          ;-
                         0152    527
                         0152    528  EXE$PAGRDERR::                                  ;PAGE READ ERROR
7E   0444 8F   3C        0152    529                    MOVZWL  #SS$_PAGRDERR,-(SP)  ;SET EXCEPTION NAME
                 05      DD   0157    530  EX5ARG: PUSHL    #5                        ;SET NUMBER OF SIGNAL ARGUMENTS
                 0033    31   0159    531                    BRW     EXE$EXCEPTION    ;FINISH IN COMMON CODE
```

EXCEPTION   - EXCEPTION HANDLING   16-SEP-1984 00:06:28 VAX/VMS Macro V04-00   Page 17
V04-000   RESERVED ADDRESSING MODE FAULT   5-SEP-1984 03:41:43 [SYS.SRC]EXCEPTION.MAR;1   (14)

C 8

```
                          015C    533                    .SBTTL  RESERVED ADDRESSING MODE FAULT
                          015C    534          ;+
                          015C    535          ; EXE$RADRMOD - RESERVED ADDRESSING MODE FAULT
                          015C    536          ;
                          015C    537          ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN AN ATTEMPT TO USE A
                          015C    538          ; RESERVED ADDRESSING MODE IS DETECTED. THE STATE OF THE STACK ON ENTRY
                          015C    539          ; IS:
                          015C    540          ;
                          015C    541          ;       00(SP) = EXCEPTION PC.
                          015C    542          ;       04(SP) = EXCEPTION PSL.
                          015C    543          ;
                          015C    544          ; THE EXCEPTION NAME FOLLOWED BY THE NUMBER OF EXCEPTION ARGUMENTS ARE
                          015C    545          ; PUSHED ON THE STACK. FINAL PROCESSING IS ACCOMPLISHED IN COMMON CODE.
                          015C    546          ;-
                          015C    547
                          015C    548                    .ALIGN  LONG
                          015C    549  EXE$RADRMOD::
     7E  044C 8F    3C    015C    550                    MOVZWL  #SS$_RADRMOD,-(SP)        ;RESERVED ADDRESSING MODE FAULT
              18    11    0161    551                    BRB     EX3ARG                    ;SET EXCEPTION NAME
```

```
                        0163    553                  .SBTTL  RESERVED OPERAND FAULT
                        0163    554        ;+
                        0163    555        ; EXE$ROPRAND - RESERVED OPERAND FAULT
                        0163    556        ;
                        0163    557        ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN AN ATTEMPT TO USE A
                        0163    558        ; RESERVED OPERAND IS DETECTED. THE STATE OF THE STACK ON ENTRY IS:
                        0163    559        ;
                        0163    560        ;       00(SP) = EXCEPTION PC.
                        0163    561        ;       04(SP) = EXCEPTION PSL.
                        0163    562        ;
                        0163    563        ; THE EXCEPTION NAME FOLLOWED BY THE NUMBER OF EXCEPTION ARGUMENTS ARE
                        0163    564        ; PUSHED ON THE STACK. FINAL PROCESSING IS ACCOMPLISHED IN COMMON CODE.
                        0163    565        ;-
                        0163    566
                        0163    567                  .ALIGN  LONG
                        0164    568        EXE$ROPRAND::                            ;RESERVED OPERAND FAULT
  7E   0454 8F   3C     0164    569                  MOVZWL  #SS$_ROPRAND,-(SP)     ;SET EXCEPTION NAME
            10   11     0169    570                  BRB     EX3ARG                 ;
```

```
                        016B    572             .SBTTL   TBIT PENDING TRAP
                        016B    573  ;+
                        016B    574  ; EXE$TBIT - TBIT PENDING TRAP
                        016B    575  ;
                        016B    576  ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN A TBIT PENDING EXCEPTION
                        016B    577  ; IS DETECTED AT THE END OF AN INSTRUCTION. THE STATE OF THE STACK ON ENTRY
                        016B    578  ; IS:
                        016B    579  ;
                        016B    580  ;         00(SP) = EXCEPTION PC.
                        016B    581  ;         04(SP) = EXCEPTION PSL.
                        016B    582  ;
                        016B    583  ; IF THE TRAP OCCURED FROM COMPATIBILITY MODE, THEN THE CODE FOR COMPATIBILITY
                        016B    584  ; MODE TBIT TRAP (7) IS PUSHED ON THE STACK AND AN EXIT IS MADE THROUGH
                        016B    585  ; THE COMPATIBILITY MODE EXCEPTION ROUTINE. ELSE THE TBIT EXCEPTION NAME
                        016B    586  ; FOLLOWED BY THE NUMBER OF EXCEPTION ARGUMENTS ARE PUSHED ON THE STACK.
                        016B    587  ; FINAL PROCESSING IS ACCOMPLISHED IN COMMON CODE.
                        016B    588  ;-
                        016B    589
                        016B    590             .ALIGN   LONG
                        016C    591  EXE$TBIT::                                      ;TBIT PENDING TRAP
              07   DD   016C    592             PUSHL    #7                          ;ASSUME TRAP FROM COMPATIBILITY MODE
   03 08 AE   1F   E1   016E    593             BBC      #PSL$V_CM,8(SP),10$         ;IF CLEAR, NOT FROM COMPATIBILITY MODE
            FF56   31   0173    594             BRW      EXE$COMPAT                  ;IF SET, GO TO COMPATIBILITY MODE CODE
   6E    045D 8F   A0   0176    595  10$:       ADDW     #SS$_TBIT-7,(SP)            ;ADJUST NAME TO THAT FOR TBIT
              03   DD   017B    596  EX3ARG:    PUSHL    #3                          ;SET NUMBER OF SIGNAL ARGUMENTS
            000F   31   017D    597             BRW      EXE$EXCEPTION               ;
```

```
                        0180    599              .SBTTL   SYSTEM SERVICE FAILURE EXCEPTION
                        0180    600    ;+
                        0180    601    ; EXE$SSFAIL - SYSTEM SERVICE FAILURE EXCEPTION
                        0180    602    ;
                        0180    603    ; THIS ROUTINE IS JUMPED TO FROM THE SYSTEM SERVICE CHANGE MODE DISPATCHER
                        0180    604    ; WHEN THE RETURN STATUS FROM A SYSTEM SERVICE INDICATES FAILURE, THE
                        0180    605    ; PREVIOUS MODE WAS USER, AND THE CURRENT PROCESS IS ENABLED FOR SYSTEM
                        0180    606    ; SERVICE FAILURE EXCEPTIONS. THE STATE OF THE STACK ON ENTRY IS:
                        0180    607    ;
                        0180    608    ;        00(SP) = CHANGE MODE PC.
                        0180    609    ;        04(SP) = CHANGE MODE PSL.
                        0180    610    ;
                        0180    611    ; WITH:
                        0180    612    ;
                        0180    613    ;        R0 = FINAL SYSTEM SERVICE STATUS.
                        0180    614    ;-
                        0180    615
                        0180    616    EXE$SSFAIL::                            ;SYSTEM SERVICE FAILURE EXCEPTION
                 50  DD 0180    617              PUSHL   R0                    ;PUSH REASON FOR SERVICE FAILURE
      7E  045C 8F  3C 0182    618              MOVZWL  #SS$_SSFAIL,-(SP)     ;SET EXCEPTION NAME
                        0187    619
                        0187    620
                        0187    621    ; THE FOLLOWING EXCEPTIONS CONVERGE TO THIS POINT:
                        0187    622    ;
                        0187    623    ;        CHANGE MODE TO SUPERVISOR,
                        0187    624    ;        CHANGE MODE TO USER, AND
                        0187    625    ;        SYSTEM SERVICE FAILURE.
                        0187    626    ;
                        0187    627
                 04  DD 0187    628    EXSSXT: PUSHL   #4                     ;SET NUMBER OF SIGNAL ARGUMENTS
      000000A6'EF  17 0189    629              JMP     EXE$REFLECT           ;REFLECT EXCEPTION TO PREVIOUS MODE
```

```
                                       018F      631                    .SBTTL  REFLECT EXCEPTION FROM MODE OTHER THAN KERNEL
                                       018F      632           ;+
                                       018F      633           ; EXE$REFLECT - REFLECT EXCEPTION FROM MODE OTHER THAN KERNEL
                                       018F      634           ;
                                       018F      635           ; THIS ROUTINE IS JUMPED TO REFLECT AN EXCEPTION FROM A MODE OTHER THAN KERNEL.
                                       018F      636           ; THE SIGNAL ARGUMENTS ARE ASSUMED TO BE SET UP PROPERLY ON THE STACK.
                                       018F      637           ; NOTE THAT THE PREVIOUS MODE FIELD OF THE PSL CONTAINS THE ACCESS MODE
                                       018F      638           ; OF THE EXCEPTION.
                                       018F      639           ;-
                                       018F      640
                                   000000A6      641                    .PSECT  YEXEPAGED1
                                       00A6      642           EXE$REFLECT::                                   ;REFLECT EXCEPTION
                             01    DD  00A6      643                    PUSHL   #1                             ;SAVE CODE INDICATING SIGNAL
                             03    BB  00A8      644                    PUSHR   #^M<R0,R1>                     ;SAVE REGISTERS R0 AND R1
                    7E       03    CE  00AA      645                    MNEGL   #3,-(SP)                       ;SET INITIAL FRAME DEPTH
                             5D    DD  00AD      646                    PUSHL   FP                             ;SET INITIAL HANDLER ESTABLISHER FRAME
                             04    DD  00AF      647                    PUSHL   #4                             ;SET NUMBER OF MECHANISM ARGUMENTS
                                       00B1      648
                                       00B1      649
                                       00B1      650           ; AT THIS POINT THE STACK HAS THE FOLLOWING FORMAT:
                                       00B1      651           ;
                                       00B1      652           ;     00(SP) = NUMBER OF MECHANISM ARGUMENTS (ALWAYS 4).
                                       00B1      653           ;     04(SP) = FP OF HANDLER ESTABLISHER FRAME (TENTATIVE).
                                       00B1      654           ;     08(SP) = FRAME DEPTH (ALWAYS -3).
                                       00B1      655           ;     12(SP) = SAVED R0.
                                       00B1      656           ;     16(SP) = SAVED R1.
                                       00B1      657           ;     20(SP) = FLAGS LONGWORD
                                       00B1      658           ;     24(SP) = NUMBER OF SIGNAL ARGUMENTS.
                                       00B1      659           ;     28(SP) = EXCEPTION NAME (INTEGER VALUE).
                                       00B1      660           ;     32(SP) = FIRST EXCEPTION PARAMETER (IF ANY).
                                       00B1      661           ;     36(SP) = SECOND EXCEPTION PARAMETER (IF ANY).
                                       00B1      662           ;                    .
                                       00B1      663           ;                    .
                                       00B1      664           ;                    .
                                       00B1      665           ;     28+N*4(SP) = N'TH EXCEPTION PARAMETER (IF ANY).
                                       00B1      666           ;     28+N*4+4(SP) = EXCEPTION PC.
                                       00B1      667           ;     28+N*4+8(SP) = EXCEPTION PSL.
                                       00B1      668
                                       00B1      669
                             50    DC  00B1      670                    MOVPSL  R0                             ;READ CURRENT PSL
              51  50  02     18    EF  00B3      671                    EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,R0,R1 ;CURRENT MODE KERNEL?
                             69    13  00B8      672                    BEQL    30$                            ;IF EQL YES
              51  50  02     16    ED  00BA      673                    CMPZV   #PSL$V_PRVMOD,#PSL$S_PRVMOD,R0,R1 ;IS CURRENT EQL PREVIOUS?
                             68    13  00BF      674                    BEQL    40$                            ;IF EQL YES
                                       00C1      675
                                       00C1      676
                                       00C1      677           ; ADJUST PREVIOUS MODE STACK POINTER USING SYSTEM SERVICE
                                       00C1      678           ;
                                       00C1      679
                    1C       BB  00C1      680                    PUSHR   #^M<R2,R3,R4>                   ;SAVE REGISTERS R2, R3, AND R4
              53  24  AE  07  C1  00C3      681                    ADDL3   #7,36(SP),R3                   ;CALCULATE NUMBER OF LONGWORDS TO MOVE
                    01EF'CF   9F  00C8      682                    PUSHAB  W^NORMAL                       ;ASSUME INFORMATION CAN BE COPIED
                    7E       D4  00CC      683                    CLRL    -(SP)                          ;SET TO USE CURRENT STACK POINTER VALUE
                    6E       9F  00CE      684                    PUSHAB  (SP)                           ;PUSH ADDRESS TO STORE UPDATED STACK VALUE
              7E  53  02     78  00D0      685                    ASHL    #2,R3,-(SP)                    ;CALCULATE STACK ADJUSTMENT VALUE
              6E  6E         CE  00D4      686                    MNEGL   (SP),(SP)                      ;SET NEGATIVE ADJUSTMENT VALUE
        7E  50  02     16    EF  00D7      687                    EXTZV   #PSL$V_PRVMOD,#PSL$S_PRVMOD,R0,-(SP) ;PUSH ACCESS MODE OF STACK
```

```
                        03  DD  00DC  688              PUSHL   #3                          ;PUSH NUMBER OF ARGUMENTS
        00000000'9F  6E  FA  00DE  689              CALLG   (SP),@#SYS$ADJSTK            ;ADJUST PREVIOUS MODE STACK POINTER
                    33  50  E8  00E5  690              BLBS    R0,20$                      ;IF LBS SUCCESSFUL COMPLETION
        14 AE   0292'CF  9E  00E8  691  10$:         MOVAB   W^BADSTACK,20(SP)           ;SET BAD STACK RETURN
            52   04 AE  D0  00EE  692              MOVL    4(SP),R2                    ;RETRIEVE PREVIOUS ACCESS MODE
    10 AE   00000000'9F42  D0  00F2  693              MOVL    @#CTL$AL_STACK[R2],16(SP)   ;SET TO USE SPECIFED STACK POINTER VALUE
        00000000'EF  6E  FA  00FB  694              CALLG   (SP),SYS$ADJSTK             ;RELOAD PREVIOUS MODE STACK POINTER
                    16  50  E8  0102  695              BLBS    R0,20$                      ;IF LBS SUCCESSFUL COMPLETION
 7E  00000000'9F42   04  C3  0105  696              SUBL3   #4,@#CTL$AL_STACK[R2],-(SP) ;CALCULATE TOP ADDRESS OF STACK RANGE
        7E   6E   0C AE  C1  010E  697              ADDL3   12(SP),(SP),-(SP)           ;CALCULATE BOTTOM ADDRESS OF STACK RANGE
                    008C  30  0113  698              BSBW    CRESTACK                    ;RE-CREATE VIRTUAL SPACE UNDER STACK
                5E   08  C0  0116  699              ADDL    #8,SP                       ;REMOVE VIRTUAL ADDRESS DESCRIPTOR
                    CD  11  0119  700              BRB     10$                         ;AND TRY AGAIN
                        011B  701
                5E   10  C0  011B  702  20$:         ADDL    #16,SP                      ;REMOVE ARGUMENT LIST FROM STACK
                    02  BA  011E  703              POPR    #^M<R1>                     ;GET NEW PREVIOUS MODE STACK POINTER VALUE
                    01EB  31  0120  704              BRW     COPYARGS                    ;
        0000019A'EF  17  0123  705  30$:         JMP     REFLECT                     ;
                    00C3  31  0129  706  40$:         BRW     NORMAL                      ;
```

```
                       012C    708                          .SBTTL   COMMON EXCEPTION EXIT
                       012C    709
                       012C    710  ; ALL EXCEPTIONS CONVERGE TO THIS POINT WITH:
                       012C    711  ;
                       012C    712  ;        00(SP) = NUMBER OF SIGNAL ARGUMENTS.
                       012C    713  ;        04(SP) = EXCEPTION NAME (INTEGER VALUE).
                       012C    714  ;        08(SP) = FIRST EXCEPTION PARAMETER (IF ANY).
                       012C    715  ;        12(SP) = SECOND EXCEPTION PARAMETER (IF ANY).
                       012C    716  ;                          .
                       012C    717  ;                          .
                       012C    718  ;
                       012C    719  ;        04+N*4(SP) = N'TH EXCEPTION PARAMETER (IF ANY).
                       012C    720  ;        04+N*4+4(SP) = EXCEPTION PC.
                       012C    721  ;        04+N*4+8(SP) = EXCEPTION PSL.
                       012C    722  ;
                       012C    723  ; NOTE THAT THE PREVIOUS MODE FIELD OF THE PSL CONTAINS THE ACCESS MODE
                       012C    724  ; OF THE EXCEPTION.
                       012C    725
                       012C    726
                   0000018F    727                          .PSECT   $AEXENONPAGED,LONG
                       018F    728                          .ENABL   LSB
                       018F    729  EXE$EXCEPTION::                                      ;THIS LABEL MUST BE GLOBAL FOR MP CODE
                 01 DD 018F    730                          PUSHL    #1                 ;SET CODE INDICATING SIGNAL
                 03 BB 0191    731                          PUSHR    #^M<R0,R1>         ;SAVE REGISTERS R0 AND R1
           7E 03 CE 0193    732                          MNEGL    #3,-(SP)           ;SET INITIAL FRAME DEPTH
              5D DD 0196    733                          PUSHL    FP                 ;SET INITIAL HANDLER ESTABLISHER FRAME
              04 DD 0198    734                          PUSHL    #4                 ;SET NUMBER OF MECHANISM ARGUMENTS
                       019A    735
                       019A    736
                       019A    737  ; AT THIS POINT THE STACK HAS THE FOLLOWING FORMAT:
                       019A    738  ;
                       019A    739  ;        00(SP) = NUMBER OF MECHANISM ARGUMENTS (ALWAYS 4).
                       019A    740  ;        04(SP) = FP OF HANDLER ESTABLISHER FRAME (TENTATIVE).
                       019A    741  ;        08(SP) = FRAME DEPTH (ALWAYS -3).
                       019A    742  ;        12(SP) = SAVED R0.
                       019A    743  ;        16(SP) = SAVED R1.
                       019A    744  ;        20(SP) = FLAGS LONGWORD
                       019A    745  ;        24(SP) = NUMBER OF SIGNAL ARGUMENTS.
                       019A    746  ;        28(SP) = EXCEPTION NAME (INTEGER VALUE).
                       019A    747  ;        32(SP) = FIRST EXCEPTION PARAMETER (IF ANY).
                       019A    748  ;        36(SP) = SECOND EXCEPTION PARAMETER (IF ANY).
                       019A    749  ;                          .
                       019A    750  ;                          .
                       019A    751  ;
                       019A    752  ;        28+N*4(SP) = N'TH EXCEPTION PARAMETER (IF ANY).
                       019A    753  ;        28+N*4+4(SP) = EXCEPTION PC.
                       019A    754  ;        28+N*4+8(SP) = EXCEPTION PSL.
                       019A    755  ;
                       019A    756
                       019A    757  REFLECT:                                            ;REFLECT EXCEPTION TO PROPER ACCESS MODE
        50 18 AE 06 C1 019A    758                          ADDL3    #6,24(SP),R0       ;CALCULATE LONGWORD OFFSET TO SAVED PSL
                       019F    759                          ASSUME   PSL$V_CM EQ 31
              6E40 D5 019F    760                          TSTL     (SP)[R0]           ;PREVIOUSLY IN COMPATIBILITY MODE?
                 20 18 01A2    761                          BGEQ     10$                ;BRANCH IF NOT
     51 00000000'9F DE 01A4    762                          MOVAL    @#CTL$AL_CMCNTX,R1 ;GET ADDRESS OF COMPATIBILITY CONTEXT AREA
           81 0C AE 7D 01AB    763                          MOVQ     12(SP),(R1)+       ;SAVE R0 AND R1
              81 52 7D 01AF    764                          MOVQ     R2,(R1)+           ;SAVE R2 AND R3
```

```
        81   54    7D  01B2   765            MOVQ    R4,(R1)+                    ;SAVE R4 AND R5
        81   56    D0  01B5   766            MOVL    R6,(R1)+                    ;SAVE R6
        81   08    9A  01B8   767            MOVZBL  #8,(R1)+                    ;SET CM EXCEPTION TYPE
     81 FC AE40    D0  01BB   768            MOVL    -4(SP)[R0],(R1)+            ;SAVE PC
        61 6E40    D0  01C0   769            MOVL    (SP)[R0],(R1)              ;SAVE PSL
             51    DC  01C4   770   10$:      MOVPSL  R1                          ;READ CURRENT PSL
        50   12    DB  01C6   771            MFPR    #PR$_IPL,R0                 ;READ CURRENT IPL
        50   02    D1  01C9   772            CMPL    #IPL$_ASTDEL,R0             ;INVALID PRIORITY LEVEL?
             04    19  01CC   773            BLSS    20$                         ;IF LSS YES
        04   51    1A  E1 01CE 774            BBC     #PSL$V_IS,R1,30$            ;IF CLR, THEN NOT ON INTERRUPT STACK
                      01D2   775   20$:      BUG_CHECK INVEXCEPTN,FATAL          ;INVALID EXCEPTION
                      01D6   776   30$:      IFNORD  #4,@#CTL$AL_STACK,20$        ;IS THERE A CONTROL REGION?
                      01E0   777                                                 ; (NOT SWAPPER,NULLPROC)
 50  51  02  16  EF 01E0  778            EXTZV   #PSL$V_PRVMOD,#PSL$S_PRVMOD,R1,R0 ;EXTRACT PREVIOUS MODE FIELD
             06    12  01E5   779            BNEQ    35$                         ;IF EQL PREVIOUS MODE WAS KERNEL
      000001EF'EF   17  01E7   780            JMP     NORMAL                      ;
      0000012C'EF   17  01ED   781   35$:     JMP     37$                         ;GOTO PAGED CODE
          0000012C      01ED   782            .PSECT  YEXEPAGED1
             1C    BB  012C   783   37$:     PUSHR   #^M<R2,R3,R4>               ;SAVE REGISTERS R2, R3, R4
        01EF'CF  9F  012E   784            PUSHAB  U^NORMAL                    ;ASSUME INFORMATION CAN BE COPIED
        52   50    D0  0132   785            MOVL    R0,R2                       ;SAVE PREVIOUS MODE
  53  28 AE   07  C1  0135   786            ADDL3   #7,40(SP),R3                ;CALCULATE NUMBER OF LONGWORDS TO MOVE
        51   52    DB  013A   787            MFPR    R2,R1                       ;READ PREVIOUS MODE STACK POINTER
          01E8    30  013D   788   38$:     BSBW    CHECK_STACK                 ;CHECK STACK RANGE
             14    12  0140   789            BNEQ    40$                         ;CONTINUE IF STACK OK
        52   03    D1  0142   790            CMPL    #PSL$C_USER,R2              ;IS IT USER MODE STACK FAULT?
             28    12  0145   791            BNEQ    50$                         ;NO, THEN BAD STACK
             0E    BB  0147   792            PUSHR   #^M<R1,R2,R3>              ;SAVE REGISTERS
        52   50    D0  0149   793            MOVL    R0,R2                       ;GET LOWEST STACK ADDRESS FROM CHECK_STACK
          02CB    30  014C   794            BSBW    EXE$EXPANDSTK               ;AND CALL TO EXPAND STACK
             0E    BA  014F   795            POPR    #^M<R1,R2,R3>              ;RESTORE REGISTERS
        1B   50    E9  0151   796            BLBC    R0,50$                      ;BR IF ANY ERROR TO DECLARE BAD STACK
             E7    11  0154   797            BRB     38$                         ;CHECK THE STACK AGAIN
                      0156   798                                                 ;NOTE- CHECK_STACK PREVENTS US FROM LOOPING
 00000000'9F42  51  D1 0156  799   40$:     CMPL    R1,@#CTL$AL_STACK[R2]        ;TOP ADDRESS OF STACK IN RANGE?
             0F    1A  015E   800            BGTRU   50$                         ;IF GTRU NO
        52   03    D1  0160   801            CMPL    #PSL$C_USER,R2              ;PREVIOUS MODE USER?
             30    13  0163   802            BEQL    70$                         ;IF EQL YES
 00000000'9F42  50  D1 0165  803            CMPL    R0,@#CTL$AL_STACKLIM[R2]     ;BOTTOM ADDRESS OF STACK IN RANGE?
             26    1E  016D   804            BGEQU   70$                         ;IF GEQU YES
      6E  0292'CF  9E  016F   805   50$:     MOVAB   U^BADSTACK,(SP)             ;SET BAD STACK RETURN
 51 00000000'9F42  D0 0174  806            MOVL    @#CTL$AL_STACK[R2],R1        ;GET STARTING ADDRESS OF PREVIOUS MODE STACK
          01A9    30  017C   807            BSBW    CHECK_STACK                 ;CHECK STACK RANGE
             14    12  017F   808            BNEQ    70$                         ;IF NEQ SUCCESSFUL RANGE CHECK
             71    DF  0181   809            PUSHAL  -(R1)                       ;PUSH TOP ADDRESS OF STACK RANGE
     7E  FC04  C1  9E  0183  810            MOVAB   -1024+4(R1),-(SP)           ;PUSH BOTTOM ADDRESS OF STACK RANGE
             18    10  0188   811            BSBB    CRESTACK                    ;RE-CREATE STACK SPACE
        5E   08    C0  018A   812            ADDL    #8,SP                       ;REMOVE VIRTUAL ADDRESS LIMITS FROM STACK
 51 00000000'9F42  D0 018D  813            MOVL    @#CTL$AL_STACK[R2],R1        ;GET STARTING ADDRESS OF PREVIOUS MODE STACK
   50  53  02  78  C2  0195  814   70$:     ASHL    #2,R3,R0                    ;CALCULATE NUMBER OF BYTES TO MOVE
        51   50    C2  0199   815            SUBL    R0,R1                       ;CALCULATE NEW TOP OF STACK ADDRESS
        52   51    DA  019C   816            MTPR    R1,R2                       ;SET NEW PREVIOUS MODE STACK POINTER
          016C    31  019F   817            BRW     COPYARGS                    ;
                      01A2   818
                      01A2   819            .DSABL  LSB
                      01A2   820
                      01A2   821   ;
```

```
                        01A2    822  ; SUBROUTINE TO CREATE VIRTUAL SPACE WHERE THE STACK SHOULD BE. TOP
                        01A2    823  ; OF STACK (UNDER THE CALL PC) IS A VA DESCRIPTOR; R2 CONTAINS ACCESS
                        01A2    824  ; MODE.
                        01A2    825  ;
                        01A2    826  ;
                        01A2    827  CRESTACK:
                        01A2    828          $SETSFM_S #0                      ;TURN OFF SYSTEM SERVICE EXCEPTIONS
        51   04 AE   9E 01AB    829          MOVAB    4(SP),R1                 ;SET ADDRESS OF VIRTUAL ADDRESS LIMITS
             50   DD 01AF    830          PUSHL    R0                       ;SAVE PREVIOUS STATE OF FAILURE MODE
                        01B1    831          $CRETVA_S (R1),(R1),R2           ;CREATE STACK PAGES FOR EXCEPTION
        09   8E   D1 01BE    832          CMPL     (SP)+,#SS$_WASSET         ;WAS SYS. SERV. FAILURE EXCEP. SET?
             0E   12 01C3    833          BNEQ     10$                      ;NO
             50   DD 01C3    834          PUSHL    R0                       ;YES, SAVE STATUS FROM $CRETVA
                        01C5    835          $SETSFM_S #1                      ;AND SET IT AGAIN
        50 8ED0 01CE    836          POPL     R0                       ;RESTORE STATUS FROM $CRETVA
        01 50   E9 01D1    837  10$:    BLBC     R0,VAFAIL                ;DEAL WITH FAILURE TO CREATE STACK
             05 01D4    838          RSB
                        01D5    839  ;
                        01D5    840  ; TO HERE ON FAILURE TO CREATE VIRTUAL ADDRESS SPACE FOR THE BASE STACK.
                        01D5    841  ; THIS ONLY OCCURRS (1) IF THE USER HAS DELETED HIS ORIGINAL STACK AND
                        01D5    842  ; THEN RUN OUT HIS VIRTUAL ADDRESS SPACE, OR (2) IF THE STACK BASE AND
                        01D5    843  ; LIMIT REGISTERS ARE SCROZZLED. ACTION DEPENDS ON THE ACCESS MODE OF
                        01D5    844  ; THE ORIGINAL EXCEPTION: FOR USER AND SUPER MODE, WE QUIETLY DELETE
                        01D5    845  ; THE PROCESS; FOR EXEC AND KERNEL MODE WE BUGCHECK NON-FATALLY AND
                        01D5    846  ; FATALLY, RESPECTIVELY.
                        01D5    847  ;
                        01D5    848  ; NOTE: R2 CONTAINS PREVIOUS ACCESS MODE
                        01D5    849  ;
        01   52   D1 01D5    850  VAFAIL: CMPL     R2,#PSL$C_EXEC           ;CHECK ACCESS MODE OF EXCEPTION
             11   1F 01D8    851          BLSSU    20$                      ;BRANCH IF KERNEL
             04   1A 01DA    852          BGTRU    10$                      ;BRANCH IF USER OR SUPER
                        01DC    853          BUG_CHECK UNABLCREVA              ;NON-FATAL FOR EXEC MODE
                        01E0    854  10$:    $DECPRC_S                         ;BYE, BYE
                        01EB    855  20$:    BUG_CHECK UNABLCREVA,FATAL        ;FATAL FOR KERNEL MODE
```

```
                                   01EF   857                .SBTTL   SEARCH FOR AND CALL CONDITION HANDLER
                                   01EF   858        ;
                                   01EF   859        ; PUSH ARGUMENT LIST ON STACK
                                   01EF   860        ;
                                   01EF   861
                                   01EF   862                .ENABL  LSB
                                   01EF   863        NORMAL:                                   ;NORMAL EXIT FROM STACK COPY
                  6E    DF         01EF   864                PUSHAL  (SP)                      ;PUSH ADDRESS OF MECHANISM ARGUMENTS
               1C AE    DF         01F1   865                PUSHAL  28(SP)                    ;PUSH ADDRESS OF SIGNAL ARGUMENTS
                  02    DD         01F4   866                PUSHL   #2                        ;PUSH NUMBER OF ARGUMENTS
                                   01F6   867
                                   01F6   868        ; CHECK IF THIS EXCEPTION SHOULD BE MODIFIED BY AN INSTRUCTION EMULATOR
                                   01F6   869
         51   00000000'EF  DO      01F6   870                MOVL    L^EXE$GL_VAXEXCVEC,R1     ;MODIFICATION ROUTINE SUPPLIED?
                  02    13         01FD   871                BEQL    EXE$SRCHANDLER            ;BRANCH IF NONE
                  61    16         01FF   872                JSB     (R1)                      ;OTHERWISE, GO THERE
                                   0201   873
                                   0201   874        ; CHECK THE PC OF THE EXCEPTION. IF IT IS IN THE CONDITION HANDLER CALL
                                   0201   875        ; VECTOR, THEN AN EXCEPTION HAS OCCURRED ATTEMPTING TO CALL A CONDITION
                                   0201   876        ; HANDLER (E.G., DUE TO BAD ADDRESS OR ENTRY MASK). IF THIS IS THE CASE,
                                   0201   877        ; EXIT THE IMAGE TO AVOID AN EXCEPTION LOOP. WE DO THE SAME FOR CALLS
                                   0201   878        ; TO AST ROUTINES. WHILE THIS IS NOT STRICTLY A BAD STACK, REPORTING THE
                                   0201   879        ; EXCEPTION WITH THE AST CONTEXT RECORDED IN THE STACK IS SUCH A PAIN
                                   0201   880        ; THAT IT IS NOT WORTH IT.  THE SPECIAL CASE OF A T-BIT PENDING EXCEPTION
                                   0201   881        ; IS ALLOWED SINCE THIS CASE CANNOT RESULT IN A LOOP.
                                   0201   882
                                   0201   883        EXE$SRCHANDLER::                          ;ENTRY POINT FOR EXTERNAL USE
         23 AE    24 AE    90      0201   884                MOVB    36(SP),35(SP)             ;SAVE SIGNAL VECTOR LENGTH
   00000464 8F    28 AE    D1      0206   885                CMPL    40(SP),#SS$_TBIT          ;CHECK FOR T-BIT PENDING EXCEPTION
                  19    13         020E   886                BEQL    10$                       ;BRANCH IF YES - SKIP CHECKS
         51   24 AE    08  C1      0210   887                ADDL3   #8,36(SP),R1              ;COMPUTE LONGWORD OFFSET TO SAVED PC
   00000000'8F   6E41    D1        0215   888                CMPL    (SP)[R1],#SYS$CALL_HANDL  ;COMPARE TO HANDLER CALL SITE
                  4E    13         021D   889                BEQL    BAD_HANDLER               ;BRANCH IF YES
   00000000'8F   6E41    D1        021F   890                CMPL    (SP)[R1],#EXE$ASTDEL      ;COMPARE TO AST CALL SITE
                  5E    13         0227   891                BEQL    BAD_AST                   ;BRANCH IF YES
                                   0229   892
                                   0229   893        ;
                                   0229   894        ; SEARCH FOR CONDITION HANDLER
                                   0229   895        ;
         0345'CF   6E    FA        0229   896        10$:    CALLG   (SP),W^SEARCH             ;SEARCH FOR CONDITION HANDLER
                  6D 50    E9      022E   897                BLBC    R0,20$                    ;IF LBC FATAL ERROR
                                   0231   898
         06 20 AE    01    E1      0231   899                BBC     #1,32(SP),11$             ;BRANCH IF THIS IS NOT A STOP
   28 AE   03   00   04    F0      0236   900                INSV    #STS$K_SEVERE,#STS$V_SEVERITY,#STS$S_SEVERITY,40(SP)
                                   023C   901                                                 ;FOR STOP, FORCE SEVERITY TO FATAL
                                   023C   902        ;
                                   023C   903        ; CALL CONDITION HANDLER
                                   023C   904        ;
                                   023C   905
         00000000'9F   16          023C   906        11$:    JSB     @#SYS$CALL_HANDL          ;CALL HANDLER VIA SYSTEM VECTOR
                  E4 50    E9      0242   907                BLBC    R0,10$                    ;IF LBC RESIGNAL
      12 20 AE    01    E0         0245   908                BBS     #1,32(SP),CONT_FROM_STOP  ;BRANCH IF ATTEMPTING TO CONTINUE FROM STOP
                                   024A   909
                                   024A   910        ; CONTINUE FROM EXCEPTION. REMOVE THE SIGNAL ARGUMENT LISTS FROM THE
                                   024A   911        ; STACK AND RESUME PROCESSING.
                                   024A   912        ;
                                   024A   913        EXE$CONTSIGNAL::
```

```
        6E   23 AE    9A  024A   914              MOVZBL   35(SP),(SP)              ;GET ORIGINAL SIGNAL ARG COUNT
             6E   08  C0  024E   915              ADDL     #8,(SP)                  ;CALCULATE LONGWORD OFFSET TO SAVED PC
             6E   04  C4  0251   916              MULL     #4,(SP)                  ;CALCULATE NUMBER OF BYTES TO REMOVE
        50   18 AE    7D  0254   917              MOVQ     24(SP),R0                ;RESTORE R0 AND R1
             5E   6E  C0  0258   918              ADDL     (SP),SP                  ;REMOVE ARGUMENT LIST FROM STACK
                      02  025B   919              RFI                              ;
                          025C   920
                          025C   921  ;
                          025C   922  ; TO HERE ON ATTEMPT TO CONTINUE FORM A CALL TO STOP
                          025C   923  ;
                          025C   924  CONT_FROM_STOP:                              ;SET FINAL STATUS AND MESSAGE
        50   00000001'8F D0  025C   925              MOVL     #<LIB$_ATTCONSTO&^C$TS$M_SEVERITY>!ATTCONSTO_IDX,R0
 FFFFFFFD 8F   14 AE    D1  0263   926              CMPL     20(SP),#-3               ;SEE IF JUST CALLED LAST CHANCE HANDLER
                  12    11  026B   927              BRB      12$                     ;AND FLOW INTO EXIT CODE
                          026D   928
                          026D   929  ;
                          026D   930  ; TO HERE IF AN EXCEPTION OCCURRED ATTEMPTING TO CALL A HANDLER
                          026D   931  ;
                          026D   932  BAD_HANDLER:
        50   28 AE    D0  026D   933              MOVL     40(SP),R0                ;SET CONDITION AS FINAL STATUS
                          0271   934                                               ;SET MESSAGE STRING
  50  03   00   02  F0  0271   935              INSV     #BADHANDLER_IDX,#STS$V_SEVERITY,#STS$S_SEVERITY,R0
 FFFFFFFD 8F   20 AE41  D1  0276   936              CMPL     32(SP)[R1],#-3           ;SEE IF TRYING TO CALL LAST CHANCE HANDLER
                  1D    12  027F   937  12$:         BNEQ     20$                     ;IF NOT, GO TO CALL IT
             20 AE   50  D0  0281   938              MOVL     R0,32(SP)               ;SAVE CONDITION AND MESSAGE
                  41    11  0285   939              BRB      30$                     ;IF YES, DON'T CALL IT AGAIN
                          0287   940
                          0287   941  ;
                          0287   942  ; TO HERE IF AN EXCEPTION OCCURRED ATTEMPTING TO CALL AN AST
                          0287   943  ;
                          0287   944  BAD_AST:
        50   28 AE    D0  0287   945              MOVL     40(SP),R0                ;SET CONDITION AS FINAL STATUS
                          028B   946                                               ;SET MESSAGE STRING
  50  03   00   03  F0  028B   947              INSV     #BADAST_IDX,#STS$V_SEVERITY,#STS$S_SEVERITY,R0
                  0C    11  0290   948              BRB      20$
                          0292   949  ;
                          0292   950  ; BAD STACK WHEN TRYING TO COPY EXCEPTION ARGUMENTS
                          0292   951  ;
                          0292   952
                          0292   953  BADSTACK:                                    ;BAD STACK EXIT FROM STACK COPY
             6E    DF  0292   954              PUSHAL   (SP)                     ;PUSH ADDRESS OF MECHANISM ARGUMENTS
             1C AE    DF  0294   955              PUSHAL   28(SP)                   ;PUSH ADDRESS OF SIGNAL ARGUMENTS
             02    DD  0297   956              PUSHL    #2                       ;PUSH NUMBER OF ARGUMENTS
                          0299   957                                               ;SET BAD STACK STATUS
        50   02B0 8F    3C  0299   958              MOVZWL   #<SS$_BADSTACK&^C$TS$M_SEVERITY>!FINAL_IDX,R0
                          029E   959
             20 AE   50  D0  029E   960  20$:         MOVL     R0,32(SP)               ;SAVE FINAL STATUS AND MESSAGE
                      02A2   961              $SETSFM_S         #0              ; CLEAR SYS. SERVICE FAILURE EXCEP. MODE
                  50  DC  02AB   962              MOVPSL   R0                       ;READ CURRENT PSL
  50  50  02  18  EF  02AD   963              EXTZV    #PSL$V_CURMOD,#PSL$S_CURMOD,R0,R0 ;EXTRACT CURRENT MODE
  51  00000000'9F40  D0  02B2   964              MOVL     @#CTL$AL_FINALEXC[R0],R1 ;GET ADDRESS OF LAST CHANCE HANDLER
                  0C    13  02BA   965              BEQL     30$                     ;IF EQL NONE
             14 AE   03  CE  02BC   966              MNEGL    #3,20(SP)               ;SET FRAME DEPTH TO MINUS THREE
  00000000'9F    16  02C0   967              JSB      @#SYS$CALL_HANDL         ;CALL LAST CHANCE CONDITION HANDLER
                  0C    11  02C6   968              BRB      35$
                          02C8   969
             50    DC  02C8   970  30$:         MOVPSL   R0                       ;READ CURRENT PSL
```

```
  50  50  02  18  EF  02CA   971              EXTZV    #PSL$V_CURMOD,#PSL$S_CURMOD,R0,R0  ;EXTRACT CURRENT MODE
          50  01  D1  02CF   972              CMPL     #PSL$C_EXEC,R0                     ;EXECUTIVE OR KERNEL MODE?
              2E  1E  02D2   973              BGEQU    45$                               ;IF GEQU YES
  50  20  AE  03  00  EF  02D4   974  35$:    EXTZV    #STS$V_SEVERITY,#STS$S_SEVERITY,32(SP),R0  ;GET MESSAGE INDEX
      00000000'9F  95  02DA   975              TSTB     @#CTL$GB_SSFILTER                 ;ARE SYSTEM SERVICES INHIBITED?
              0E  12  02E0   976              BNEQ     40$                               ;YES, DON'T TRY TO PRINT ANYTHING
              6E  9F  02E2   977              PUSHAB   (SP)                              ;PUSH ADDRESS OF CONDITION ARGUMENT LIST
      FD17 CF40  DD  02E4   978              PUSHL    MSG_VECTOR[R0]                    ;PUSH ADDRESS OF FINAL EXCEPTION MESSAGE
  00000000'EF  02  FB  02E9   979              CALLS    #2,EXE$EXCMSG                     ;PRINT FINAL EXCEPTION MESSAGE
      50  20  AE  D0  02F0   980  40$:    MOVL     32(SP),R0                        ;RETRIEVE FINAL STATUS
  50  03  00  04  F0  02F4   981              INSV     #STS$K_SEVERE,#STS$V_SEVERITY,#STS$S_SEVERITY,R0
                  02F9   982              $EXIT_S  R0                               ;EXIT PROCESS
                  0302   983
              06  1A  0302   984  45$:    BGTRU    50$                              ;BRANCH IF KERNEL MODE
                  0304   985              BUG_CHECK FATALEXCPT                       ;FATAL EXECUTIVE MODE EXCEPTION
              E6  11  0308   986              BRB      40$                              ;GO DELETE THE PROCESS
                  030A   987
                  030A   988  50$:    BUG_CHECK FATALEXCPT,FATAL                  ;FATAL KERNEL MODE EXCEPTION
                  030E   989
                  030E   990              .DSABL   LSB
                  030E   991
                  030E   992  ; COPY ARGUMENTS TO PREVIOUS MODE STACK AND EXIT TO PREVIOUS MODE
                  030E   993  ;
                  030E   994  ;
                  030E   995
                  030E   996  COPYARGS:                                            ;COPY ARGUMENT LISTS TO PREVIOUS MODE STACK
      50  10  AE  9E  030E   997              MOVAB    16(SP),R0                        ;GET ADDRESS OF ARGUMENTS TO COPY
          81  80  D0  0312   998  10$:    MOVL     (R0)+,(R1)+                      ;COPY EXCEPTION ARGUMENTS TO PREVIOUS STACK
          FA  53  F5  0315   999              SOBGTR   R3,10$                           ;ANY MORE LONGWORDS TO COPY?
  70  C8000010  8F  CA  0318  1000              BICL     #PSL$M_CM!PSL$M_TBIT!-           ;CLEAR COMPATIBILITY, T-BIT, T PENDING, AND
              031F  1001                       PSL$M_FPD!PSL$M_TP,-(R0)         ;FIRST PART DONE
          70  8ED0  031F  1002              POPL     -(R0)                            ;SET RETURN ADDRESS
              1C  BA  0322  1003              POPR     #^M<R2,R3,R4>                    ;RESTORE REGISTERS R2, R3, R4
      5E  50  D0  0324  1004              MOVL     R0,SP                            ;REMOVE ARGUMENTS FROM KERNAL STACK
              02  0327  1005              REI                                      ;
                  0328  1006
                  0328  1007
                  0328  1008  ; SUBROUTINE TO CHECK ACCESSIBILITY OF STACK ADDRESS RANGE
                  0328  1009  ;
                  0328  1010  ; INPUTS:
                  0328  1011  ;       R1 - STACK POINTER
                  0328  1012  ;       R3 - PARTIAL LONGWORD COUNT
                  0328  1013  ;
                  0328  1014  ; OUTPUTS:
                  0328  1015  ;       R0 - BOTTOM ADDRESS OF RANGE
                  0328  1016  ;       Z CONDITION CODE - 0 IF ACCESSIBLE, ELSE 1
                  0328  1017  ;
                  0328  1018  ;       R1,R2,R3 ARE PRESERVED.
                  0328  1019  ;
                  0328  1020  CHECK_STACK:                                         ;CHECK STACK ADDRESS RANGE
              0E  BB  0328  1021              PUSHR    #^M<R1,R2,R3>                    ;SAVE REGISTERS
      51  53  15  C1  032A  1022              ADDL3    #3+1+17,R3,R1                    ;CALCULATE TOTAL LONGWORDS IN RANGE
          51  04  C4  032E  1023              MULL     #4,R1                            ;CALCULATE NUMBER OF BYTES IN RANGE
      50  6E  51  C3  0331  1024              SUBL3    R1,(SP),R0                       ;CALCULATE BOTTOM ADDRESS OF RANGE
              50  DD  0335  1025              PUSHL    R0                               ;SAVE THIS QUANTITY
              53  D4  0337  1026              CLRL     R3                               ;ACCESS MODE TO MAXIMIZE WITH PSL<PRVMOD>
      00000000'EF  16  0339  1027              JSB      EXE$PROBEW                       ;CHECK WRITE ACCESS
```

EXCEPTION   - EXCEPTION HANDLING   16-SEP-1984 00:06:28 VAX/VMS Macro V04-00   Page 29
V04-000   SEARCH FOR AND CALL CONDITION HANDLER   5-SEP-1984 03:41:43 [SYS.SRC]EXCEPTION.MAR;1   (20)

                                                            B 9

```
       50   01   D3  033F  1028        BITL    #1,R0                    ;SET CONDITION CODE
            0F   BA  0342  1029        POPR    #^M<R0,R1,R2,R3>         ;RESTORE REGISTERS (NOTE: COND. CODES
                     0344  1030                                        ;  PRESERVED
            05       0344  1031        RSB                             ;RETURN
```

```
                              0345  1033                .SBTTL  SEARCH FOR CONDITION HANDLER
                              0345  1034
                              0345  1035     ; SEARCH - SEARCH FOR CONDITION HANDLER
                              0345  1036     ;
                              0345  1037     ; THIS IS A SPECIAL INTERNAL ROUTINE THAT IS CALLED IN THE INITIAL SEARCH
                              0345  1038     ; FOR A CONDITION HANDLER AND ON RESIGNAL FROM A PREVIOUSLY SIGNALLED
                              0345  1039     ; CONDITION.
                              0345  1040     ;
                              0345  1041     ; INPUTS:
                              0345  1042     ;
                              0345  1043     ;          00(AP) = NUMBER OF CONDITION ARGUMENTS.
                              0345  1044     ;          04(AP) = ADDRESS OF SIGNAL ARGUMENT LIST.
                              0345  1045     ;          08(AP) = ADDRESS OF MECHANISM ARGUMENT LIST.
                              0345  1046     ;          12(AP) = NUMBER OF MECHANISM ARGUMENTS.
                              0345  1047     ;          16(AP) = FP OF HANDLER ESTABLISHER FRAME.
                              0345  1048     ;          20(AP) = FRAME DEPTH.
                              0345  1049     ;          24(AP) = SAVED R0.
                              0345  1050     ;          28(AP) = SAVED R1.
                              0345  1051     ;          32(AP) = FLAGS LONGWORD
                              0345  1052     ;          36(AP) = NUMBER OF SIGNAL ARGUMENTS.
                              0345  1053     ;          40(AP) = EXCEPTION NAME (INTEGER VALUE).
                              0345  1054     ;          44(AP) = FIRST EXCEPTION PARAMETER (IF ANY).
                              0345  1055     ;          48(AP) = SECOND EXCEPTION PARAMETER (IF ANY).
                              0345  1056     ;                .
                              0345  1057     ;                .
                              0345  1058     ;                .
                              0345  1059     ;          40+N*4(AP) = N'TH EXCEPTION PARAMETER (IF ANY).
                              0345  1060     ;          40+N*4+4(AP) = EXCEPTION PC.
                              0345  1061     ;          40+N*4+8(AP) = EXCEPTION PSL.
                              0345  1062     ;
                              0345  1063     ; OUTPUTS:
                              0345  1064     ;
                              0345  1065     ;     R0 LOW BIT CLEAR INDICATES FAILURE TO LOCATE CONDITION HANDLER.
                              0345  1066     ;
                              0345  1067     ;          R0 = SS$_ACCVIO - STACK CANNOT BE READ FROM CURRENT MODE.
                              0345  1068     ;
                              0345  1069     ;          R0 = SS$_NOHANDLER - NO CONDITION HANDLER COULD BE FOUND.
                              0345  1070     ;
                              0345  1071     ;     R0 LOW BIT SET INDICATES SUCCESSFUL COMPLETION.
                              0345  1072     ;
                              0345  1073     ;          R1 = ADDRESS OF CONDITION HANDLER.
                              0345  1074     ;
                              0345  1075     ;
                              0345  1076                .ENABLE LSB
                              0345  1077
                              0345  1078  SEARCH:                                         ;SEARCH FOR CONDITION HANDLER
                        0000  0345  1079                .WORD   0                         ;ENTRY MASK
        6D   03F3'CF  9E  0347  1080                MOVAB   W^EXE$SIGTORET,(FP)       ;SET ADDRESS OF CONDITION HANDLER
              50   10 AC  D0  034C  1081  10$:        MOVL    16(AP),R0                 ;GET PREVIOUS FRAME ADDRESS
                    51  DC  0350  1082  20$:        MOVPSL  R1                        ;READ CURRENT PSL
     51   51   02   18  EF  0352  1083                EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,R1,R1  ;EXTRACT CURRENT MODE
                 14 AC  D6  0357  1084                INCL    20(AP)                    ;INCREMENT FRAME DEPTH
                    37   13  035A  1085                BEQL    50$                       ;IF EQL FIRST STACK FRAME
                    1B   14  035C  1086                BGTR    40$                       ;IF GTR OTHER STACK FRAME
     50   00000000'9F41  7E  035E  1087                MOVAQ   @#CTL$AQ_EXCVEC[R1],R0    ;GET ADDRESS OF EXCEPTION VECTOR QUADWORD
  14 AC   FFFFFFFE 8F  D1  0366  1088                CMPL    #-2,20(AP)                ;EXAMINE PRIMARY VECTOR?
                    02   13  036E  1089                BEQL    30$                       ;IF EQL YES
```

```
                        80    D5  0370  1090          TSTL    (RO)+                      ;ADJUST TO SECONDARY VECTOR
                  51    60    DO  0372  1091  30$:     MOVL    (RO),R1                    ;GET ADDRESS OF CONDITION HANDLER
                        23    12  0375  1092          BNEQ    60$                        ;IF NEQ CONDITION HANDLER FOUND
                        D3    11  0377  1093          BRB     10$
                  70 16 AC    E8  0379  1094  40$:     BLBS    22(AP),100$                ;IF LBS SEARCH COUNT OVERFLOW
                        4D    10  037D  1095  45$:     BSBB    CHECK_FP                   ;CHECK IF THIS FRAME IS VALID
        10 A0  00000004'8F    D1  037F  1096          CMPL    #SYS$CALL_HANDL+4,SAVPC(RO) ;CALL FROM CONDITION DISPATCHER?
                        15    13  0387  1097          BEQL    70$                        ;BRANCH IF YES - MUST SKIP FRAMES
                  50  0C A0    DO  0389  1098          MOVL    SAVFP(RO),RO               ;GET ADDRESS OF PREVIOUS FRAME
                        5E    13  038D  1099          BEQL    100$                       ;IF EQL NONE
                  10 AC  50    DO  038F  1100  48$:     MOVL    RO,16(AP)                  ;SAVE ADDRESS OF ESTABLISHER FRAME
                        37    10  0393  1101  50$:     BSBB    CHECK_FP                   ;CHECK IF THIS FRAME IS VALID
                  51    60    DO  0395  1102          MOVL    (RO),R1                    ;GET ADDRESS OF CONDITION HANDLER
                        B2    13  0398  1103          BEQL    10$                        ;IF EQL NONE
                  50    01    C8  039A  1104  60$:     BISL    #1,RO                      ;INDICATE SUCCESSFUL COMPLETION
                        04        039D  1105          RET                                ;
                              039E  1106
   51  06 A0  0C  00    EF  039E  1107  70$:     EXTZV   #0,#12,SAVMSK(RO),R1       ;GET REGISTER SAVE MASK
   7E  06 A0  02  0E    EF  03A4  1108          EXTZV   #14,#2,SAVMSK(RO),-(SP)    ;GET STACK ALIGNMENT BIAS
                  50    14    C0  03AA  1109          ADDL    #SAVRG,RO                  ;ADD OFFSET TO REGISTER SAVE AREA
                  50    8E    C0  03AD  1110          ADDL    (SP)+,RO                   ;ADD STACK ALIGNMENT BIAS
                  03 51    E9  03B0  1111  80$:     BLBC    R1,90$                     ;IF LBC CORRESPONDING REGISTER NOT SAVED
                  50    04    C0  03B3  1112          ADDL    #4,RO                      ;ADJUST FOR SAVED REGISTER
           51  51  FF    8F    78  03B6  1113  90$:     ASHL    #-1,R1,R1                  ;ANY MORE REGISTERS SAVED?
                        F3    12  03BB  1114          BNEQ    80$                        ;IF NEQ YES
                  51  0C A0    DO  03BD  1115          MOVL    CHF$L_MCHARGLST+4(RO),R1   ;GET ADDRESS OF MECHANISM ARGUMENTS
                  50  04 A1    DO  03C1  1116          MOVL    CHF$L_MCH_FRAME(R1),RO     ;GET ADDRESS OF ESTABLISHER FRAME
                        08 A1    D5  03C5  1117          TSTL    CHF$L_MCH_DEPTH(R1)        ;CHECK IF THIS IS A VECTORED HANDLER
                        C5    19  03C8  1118          BLSS    48$                        ;IF SO, DON'T SKIP 'ESTABLISHER'
                        B1    11  03CA  1119          BRB     45$                        ;
                              03CC  1120
                              03CC  1121  ;
                              03CC  1122  ; SUBROUTINE TO VALIDATE THE CURRENT FRAME ADDRESS. THIS IS DONE WITH
                              03CC  1123  ; A RANGE CHECK AGAINST THE STACK LIMIT REGISTERS IN THE P1 VECTOR PAGE.
                              03CC  1124  ; SINCE FP LINKAGES EXTEND ACROSS ACCESS MODES, THERE IS NO OTHER CHECK
                              03CC  1125  ; POSSIBLE TO PREVENT CHASING AN INNER MODE EXCEPTION OUT TO AN OUTER
                              03CC  1126  ; ACCESS MODE.
                              03CC  1127  ;
                              03CC  1128  CHECK_FP:
                        51    DC  03CC  1129          MOVPSL  R1                         ;READ CURRENT PSL
           51  51  02    18    EF  03CE  1130          EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,R1,R1 ;EXTRACT CURRENT MODE
   00000000'9F41  50    D1  03D3  1131          CMPL    RO,@#CTL$AL_STACK[R1]      ;FRAME POINTER WITHIN STACK RANGE?
                        10    1A  03DB  1132          BGTRU   100$                       ;IF GTRU NO
                        51    03    D1  03DD  1133          CMPL    #PSL$C_USER,R1             ;IF IN USER MODE
                        0A    13  03E0  1134          BEQL    95$                        ;SKIP TOP RANGE CHECK
   00000000'9F41  50    D1  03E2  1135          CMPL    RO,@#CTL$AL_STACKLIM[R1]   ;FRAME POINTER WITHIN STACK RANGE?
                        01    1F  03EA  1136          BLSSU   100$                       ;IF LSSU NO
                        05        03EC  1137  95$:     RSB                                ;STACK FRAME OK
                  50  08F8 8F    3C  03ED  1138  100$:    MOVZWL  #SS$_NOHANDLER,RO          ;SET NO HANDLER FOUND
                        04        03F2  1139          RET                                ;
                              03F3  1140
                              03F3  1141          .DISABLE LSB
```

EXCEPTION
V04-000

E 9
- EXCEPTION HANDLING   16-SEP-1984 00:06:28 VAX/VMS Macro V04-00   Page 32
EXE$SIGTORET - TURN EXCEPTION INTO RETUR 5-SEP-1984 03:41:43 [SYS.SRC]EXCEPTION.MAR;1   (22)

```
                                    03F3  1143              .SBTTL  EXE$SIGTORET - TURN EXCEPTION INTO RETURN STATUS
                                    03F3  1144      ;++
                                    03F3  1145      ; FUNCTIONAL DESCRIPTION:
                                    03F3  1146      ;
                                    03F3  1147      ;     THIS IS A CONDITION HANDLER THAT TURNS AN EXCEPTION IN THE ESTABLISHER
                                    03F3  1148      ;     FRAME INTO A RETURN FROM THE ESTABLISHER FRAME WITH THE EXCEPTION NAME
                                    03F3  1149      ;     AS THE STATUS.  EXCEPTIONS FROM ANY FRAME OTHER THAN THE ESTABLISHER
                                    03F3  1150      ;     ARE RESIGNALLED.  UNWINDS ARE IGNORED.
                                    03F3  1151      ;
                                    03F3  1152      ; INPUT PARAMETERS:
                                    03F3  1153      ;     00(AP) = NUMBER OF CONDITION ARGUMENTS.
                                    03F3  1154      ;     04(AP) = ADDRESS OF SIGNAL ARGUMENT LIST.
                                    03F3  1155      ;     08(AP) = ADDRESS OF MECHANISM ARGUMENT LIST.
                                    03F3  1156      ;
                                    03F3  1157      ; OUTPUT PARAMETERS:
                                    03F3  1158      ;     R0 - COMPLETION STATUS CODE
                                    03F3  1159      ;          SS$_RESIGNAL - ALWAYS
                                    03F3  1160      ;
                                    03F3  1161      ;
                                    03F3  1162      EXE$SIGTORET::
                              0000  03F3  1163              .WORD   0
                                    03F5  1164      ;
                                    03F5  1165              ASSUME  CHF$L_MCHARGLST,EQ,CHF$L_SIGARGLST+4
                   50   04 AC  7D   03F5  1166              MOVQ    CHF$L_SIGARGLST(AP),R0  ; GET ADDRESS OF SIGNAL ARGUMENT LIST
      04 A0  00000920 8F   D1       03F9  1167              CMPL    #SS$_UNWIND,CHF$L_SIG_NAME(R0) ; UNWINDING?
                         11   13    0401  1168              BEQL    10$                     ; BRANCH TO EXIT IF YES
                      08 A1   D5    0403  1169              TSTL    CHF$L_MCH_DEPTH(R1)     ; EXCEPTION WITHIN ESTABLISHER FRAME?
                         0C   12    0406  1170              BNEQ    10$                     ; BRANCH AND RETURN RESIGNAL IF NO
      0C A1   04 A0   D0    0408  1171              MOVL    CHF$L_SIG_NAME(R0),CHF$L_MCH_SAVR0(R1) ; SET RETURN STATUS
                      7E   7C    040D  1172              CLRQ    -(SP)                   ; CLEAR DEPTH AND NEW PC ARGUMENTS
           0000'CF   02   FB    040F  1173              CALLS   #2,W^EXE$UNWIND         ; UNWIND TO ESTABLISHER'S CALLER
        50   0918 8F   3C    0414  1174  10$:         MOVZWL  #SS$_RESIGNAL,R0        ; RETURN RESIGNAL STATUS
                         04    0419  1175              RET                             ;
```

EXCEPTION
V04-000
                    - EXCEPTION HANDLING   16-SEP-1984 00:06:28   VAX/VMS Macro V04-00   Page 33
                    EXE$EXPANDSTK - EXPAND USER STACK   5-SEP-1984 03:41:43 [SYS.SRC]EXCEPTION.MAR;1   (23)

F 9

```
                              041A  1177                  .SBTTL  EXE$EXPANDSTK - EXPAND USER STACK
                              041A  1178          ;++
                              041A  1179          ; FUNCTIONAL DESCRIPTION:
                              041A  1180          ;     EXPAND STACK IS CALLED BY EXE$ACVIOLAT,  REFLECT, AND SYS$ADJSTK
                              041A  1181          ;     TO ALLOCATE MORE SPACE TO THE USER STACK.
                              041A  1182          ;
                              041A  1183          ; INPUT PARAMTERS:
                              041A  1184          ;     R2 - VIRTUAL ADDRESS LOWER BOUND FOR STACK.
                              041A  1185          ;
                              041A  1186          ; OUTPUT PARAMETERS:
                              041A  1187          ;     R0 - COMPLETION STATUS CODE
                              041A  1188          ;          SS$_NORMAL - SUCCESSFUL COMPLETION
                              041A  1189          ;          SS$_VASFULL -VIRTUAL ADDRESS SPACE FULL
                              041A  1190          ;          SS$_PAGOWNVIO - PAGE OWNER VIOLATION
                              041A  1191          ;          SS$_EXQUOTA - PAGING FILE QUOTA EXCEEDED
                              041A  1192          ;          SS$_INSFWSL - INSUFFICIENT WORKING SET SIZE
                              041A  1193          ;--
                              041A  1194          EXE$EXPANDSTK::
                      3E  BB  041A  1195                  PUSHR   #^M<R1,R2,R3,R4,R5>      ; SAVE WORKING REGISTERS
54  00000000'9F  D0  041C  1196                  MOVL    @#CTL$GL_PCB,R4          ; GET PCB ADDRESS
0000000C'9F  52  D1  0423  1197                  CMPL    R2,@#CTL$AL_STACK+12     ; IS ADDRESS IN USER STACK?
                  5E  1A  042A  1198                  BGTRU   50$                     ; NO, DEFINITE ACCESS VIOLATION
         5A 52  1E  E1  042C  1199                  BBC     #VA$V_P1,R2,50$         ; BR IF NOT P1 SPACE ADDRESS
55  00000000'9F  D0  0430  1200                  MOVL    @#CTL$GL_PHD,R5          ; GET PHD ADDRESS
    00000000'EF  16  0437  1201                  JSB     MMG$PTEINDX             ; GET LW INDEX TO PTE IN PHD
             15 50  E9  043D  1202                  BLBC    R0,20$                  ; BR IF JUST EXPAND REGION
         6C B443  D5  0440  1203                  TSTL    @PCB$L_PHD(R4)[R3]      ; CHECK PTE OF ACCVIO VA
                  44  12  0444  1204                  BNEQ    50$                     ; BR IF NOT EMPTY
                  53  D6  0446  1205          10$:    INCL    R3                      ; NEXT PTE INDEX
         6C B443  D5  0448  1206                  TSTL    @PCB$L_PHD(R4)[R3]      ; CHECK PTE FOR EMPTY
                  0B  12  044C  1207                  BNEQ    30$                     ; NO, STOPPER FOR CREATE
    52  0200 C2  9E  044E  1208                  MOVAB   512(R2),R2              ; AUGMENT VA BY ANOTHER PAGE
                  F1  11  0453  1209                  BRB     10$                     ; AND TRY ANOTHER PAGE
    52    30 A5  D0  0455  1210          20$:    MOVL    PHD$L_FREP1VA(R5),R2    ; SET ENDING ADDRESS TO FREP1VA
             04 AE  DD  0459  1211          30$:    PUSHL   4(SP)                   ; BUILD INADR DESCRIPTOR
                  52  DD  045C  1212                  PUSHL   R2
             53  7E  7E  045E  1213                  MOVAQ   -(SP),R3                ; CREATE SPACE FOR RETADR
                      0461  1214                  $CRETVA_S 8(R3),(R3),#PSL$C_USER; CREATE SPACE
             13 50  E8  046F  1215                  BLBS    R0,40$                  ; BR IF SUCCESS
             50  DD  0472  1216                  PUSHL   R0                      ; SAVE ERROR STATUS
                      0474  1217                  $DELTVA_S (R3),8(R3),#PSL$C_USER; DELETE PARTIAL AREA
         50 8ED0  0482  1218                  POPL    R0                      ; RESTORE COMPLETION CODE
             5E  10  C0  0485  1219          40$:    ADDL    #16,SP                  ; CLEAN STACK
                  03  11  0488  1220                  BRB     60$
         50  0C  3C  048A  1221          50$:    MOVZWL  #SS$_ACCVIO,R0          ; SET ERROR STATUS
                  3E  BA  048D  1222          60$:    POPR    #^M<R1,R2,R3,R4,R5>     ; RESTORE REGISTERS
                      05  048F  1223                  RSB                             ; AND RETURN
                      0490  1224
                      0490  1225
```

```
                         0490  1227                    .SBTTL   EXE$MCHK_PRTCT - MACHINE CHECK RECOVERY BLOCK
                         0490  1228
                         0490  1229          ;++
                         0490  1230          ; EXE$MCHK_PRTCT -ENABLE RECOVERY BLOCK FOR MACHINE CHECK EXCEPTIONS
                         0490  1231          ;
                         0490  1232          ; FUNCTIONAL DESCRIPTION:
                         0490  1233          ;
                         0490  1234          ;       ALLOW A SPECIFIED BLOCK OF KERNEL CODE TO PROTECT ITS SELF
                         0490  1235          ;       FROM FATAL MACHINE CHECKS, THEN FIND OUT IF ONE OCCURED WITHING
                         0490  1236          ;       THE EXECUTION OF THE BLOCK.
                         0490  1237          ;
                         0490  1238          ; INPUTS:
                         0490  1239          ;
                         0490  1240          ;       R0 = FUNCTION MASK TO FILTER SPECIFIC TYPES OF MACHINE CHECKS
                         0490  1241          ;       (SP) = RETURN ADDRESS (START OF RECOVERY BLOCK)
                         0490  1242          ;       4(SP) = END OF RECOVERY BLOCK ADDRESS
                         0490  1243          ;       MUST BE IN KERNEL MODE
                         0490  1244          ;       CODE IN RECOVERY BLOCK EXECUTES AT IPL 31
                         0490  1245          ;
                         0490  1246          ; OUTPUTS:
                         0490  1247          ;
                         0490  1248          ;       WHEN INSTRUCTION AFTER END OF RECOVERY BLOCK IS REACHED,
                         0490  1249          ;               R0 = 0 IF MACHINE CHECK OCCURRED
                         0490  1250          ;               R1 = 1 IF MACHINE CHECK DID NOT OCCUR
                         0490  1251          ;       ALL OTHER REGISTERS PRESERVED
                         0490  1252          ;
                         0490  1253          ; CALLING SEQUENCE:
                         0490  1254          ;
                         0490  1255          ;       PUSHAL   END_LABEL              ; LABEL, END OF RECOVERY BLOCK ON STACK
                         0490  1256          ;       MOVL     #MASK,R0               ; FUNCTION FILTER MASK
                         0490  1257          ;       JSB      EXE$MCHK_PRTCT         ; INITIATE RECOVERY BLOCK
                         0490  1258          ;                                       ; PROTECTED CODE, EXECUTED AT IPL 31
                         0490  1259          ;               .
                         0490  1260          ;               .
                         0490  1261          ;       RSB                            ; END OF RECOVERY BLOCK RETURN
                         0490  1262          ; END_LABEL:                           ; END OF RECOVERY BLOCK LABEL
                         0490  1263          ;       BLBS     R0,MCHK_OK            ; IF LBS, NO MACHINE CHECK OCCURED
                         0490  1264          ;       BRW      MCHK_ERROR           ; ELSE, CODE FAULTED
                         0490  1265          ;--
                         0490  1266
                      000001F3  1267                    .PSECT  $AEXENONPAGED,LONG
                         01F3  1268
                         01F3  1269          EXE$MCHK_PRTCT::
                         01F3  1270
  7E    0000'CF   7D    01F3  1271                    MOVQ    W^MCHK$GL_MASK,-(SP)  ; SAVE PREVIOUS MASK AND SP - RECURSIVE
                         01F8  1272                                                 ; SO MACHINE CHECK HANDLER CAN USE PRTCT
                         01F8  1273                    DSBINT                       ; GO TO IPL 31 FOR REMAINDER OF BLOCK
  0000'CF   50    D0    01FE  1274                    MOVL    R0,W^MCHK$GL_MASK    ; CURRENT FUNCTION MASK
  0000'CF   5E    D0    0203  1275                    MOVL    SP,W^MCHK$GL_SP      ; SAVE CURRENT SP FOR POTENTIAL RECOVERY
            0C BE  16    0208  1276                    JSB     @<$*4>(SP)           ; CALL PROTECTED CODE BACK
                         020B  1277
                         020B  1278          ; IF PROTECTED CODE EXECUTED WITHOUT A MACHINE CHECK, IT RETURNS HERE
                         020B  1279
      50   01   3C      020B  1280                    MOVZWL  #SS$_NORMAL,R0       ; NORMAL COMPLETION, NO MACHINE CHECK
                         020E  1281
                         020E  1282          ; RESTORE MASK AND SP OF OLD, FINAL RETURN TO CALLER
                         020E  1283
```

```
                              020E  1284 PROTECT_RETURN:                                    ; COMMON RETURN TO END OF RECOVERY BLOCK
                              020E  1285
 0000'CF   04 AE  7D 020E    1286                   MOVQ   <1*4>(SP),W^MCHK$GL_MASK ; RESTORE PREVIOUS MASK AND SP
                              0214  1287                   ENBINT                          ; BACK TO ORIGONAL IPL
         5E   0C  CO 0217    1288                   ADDL   #<3*4>,SP                       ; CLEAN UP STACK
                  05 021A    1289                   RSB                                    ; FINAL RETURN TO CALLER AT END OF
                     021B    1290                                                          ; RECOVERY BLOCK LABEL
```

EXCEPTION
V04-000

I 9

- EXCEPTION HANDLING   16-SEP-1984 00:06:28   VAX/VMS Macro V04-00   Page 36
EXE$MCHK_PRTCT - MACHINE CHECK RECOVERY   5-SEP-1984 03:41:43   [SYS.SRC]EXCEPTION.MAR;1   (25)

```
                              021B  1292 ;++
                              021B  1293 ;
                              021B  1294 ; EXE$MCHK_BUGCHK - HANDLE ALL BUGCHECKS FOR WHICH PROTECTION IS DESIRED
                              021B  1295 ;
                              021B  1296 ; FUNCTIONAL DESCRIPTION:
                              021B  1297 ;
                              021B  1298 ;      THIS REQUTINE IS CALLED FROM WITHIN MACHINE CHECK HANDLER JUST
                              021B  1299 ;      BEFORE ISSUING A FATAL BUG-CHECK.  IF A CURRENT PROTECTION BLOCK
                              021B  1300 ;      IS IN EFFECT, A RETURN IS MADE AT THE END OF RECOVERY BLOCK LABEL
                              021B  1301 ;      AFTER PROTECTED CODE.  ELSE, IF NO PROTECTION BLOCK IN EFFECT, RETURN
                              021B  1302 ;      TO MAHCINE CHECK WHICH ISSUES THE FATAL BUG-CHECK.
                              021B  1303 ;
                              021B  1304 ; INPUTS:
                              021B  1305 ;
                              021B  1306 ;      (SP) = RETURN ADDRESS TO MACHINE CHECK
                              021B  1307 ;      4(SP) = ADDRESS OF MACHINE CHECK PC,PSL FROM EXCEPTION
                              021B  1308 ;      8(SP) = FUNCTION FILTER MASK BUILT BY MACHINE CHECK, DESCRIBES THE
                              021B  1309 ;              TYPE OF MACHINE CHECK FOR TESTING AGIANST MASK SPECIFIED BY
                              021B  1310 ;              PROTECTED CODE.
                              021B  1311 ;
                              021B  1312 ; OUTPUTS:
                              021B  1313 ;
                              021B  1314 ;      IF NO RECOVERY BLOCK IN EFFECT:
                              021B  1315 ;
                              021B  1316 ;              PC,PSL POINTER AND MASK REMOVED FROM STACK
                              021B  1317 ;              RSB BACK TO MACHINE CHECK
                              021B  1318 ;              ALL REGISTERS PRESERVED
                              021B  1319 ;
                              021B  1320 ;      IF RECOVERY BLOCK IN EFFECT:
                              021B  1321 ;
                              021B  1322 ;              MODE, IPL, STACK SET TO THAT IN EFFECT WHEN RECOVERY BLOCK
                              021B  1323 ;                      DECLARED.
                              021B  1324 ;              R0 - CODE INDICATING MACHINE CHECK OCCURED
                              021B  1325 ;              RETURN TO END OF RECOVERY BLOCK LABEL
                              021B  1326 ;              ALL OTHER REGISTERS PRESERVED
                              021B  1327 ;
                              021B  1328 ;--
                              021B  1329
                              021B  1330 EXE$MCHK_BUGCHK::
                              021B  1331
                    07  BB    021B  1332        PUSHR  #^M<R0,R1,R2>          ; SAVE SOME REGISTERS
         51    10 AE  7D      021D  1333        MOVQ   <4+4>(SP),R1           ; GET PC,PSL POINTER AND MASK
               2F   10        0221  1334        BSBB   MCHK_TEST             ; CURRENTLY A RECOVERY BLOCK?
            0A 50   E8        0223  1335        BLBS   R0,15$                ; BRANCH IF YES
                    07  BA    0226  1336 10$:   POPR   #^M<R0,R1,R2>          ; NO RECOVERY BLOCK, RESTORE REGS
         04 AE  8E  D0        0228  1337        MOVL   (SP)+,4(SP)           ; MOVE RETURN ADDRESS UP ON STACK
            5E  04  C0        022C  1338        ADDL   #<4+1>,SP             ; CLEAR STACK OF MASK
                    05        022F  1339        RSB                          ; RETURN TO MACHINE CHECK HANDLER
                              0230  1340
         52   01  8A          0230  1341 15$:   BICB   #MCHK$M_LOG,R2         ; IGNORE LOG INHIBIT BIT
   0000'CF  52  D3            0233  1342        BITL   R2,W^MCRK$GL_MASK      ; PROTECTION DESIRED HERE?
               EC  13        0238  1343        BEQL   10$                    ; NO, RETURN TO MACHINE CHECK
                    07  BA   023A  1344        POPR   #^M<R0,R1,R2>          ; RESTORE REGISTERS
         5E  04 AE  D0       023C  1345        MOVL   4(SP),SP               ; RESET STACK TO INTERRUPT PC,PSL
   6E  0246'CF  DE           0240  1346        MOVAL  W^20$,(SP)             ; REI BACK HERE INSTEAD
               02            0245  1347        REI                          ; BACK TO PROTECTED CODES' MODE, ETC.
   50   02BC 8F  3C          0246  1348 20$:   MOVZWL #SS$_MCHECK,R0         ; ERROR CODE IN R0
```

```
5E   0000'CF  D0  024B  1349        MOVL    W^MCHK$GL_SP,SP         ; RESET STACK FOR RETURN
     BC       11  0250  1350        BRB     PROTECT_RETURN         ; RETURN TO END OF PROTECTED CODE
```

EXCEPTION
V04-000

K 9

- EXCEPTION HANDLING   16-SEP-1984 00:06:28 VAX/VMS Macro V04-00   Page 38
EXE$MCHK_PRTCT - MACHINE CHECK RECOVERY 5-SEP-1984 03:41:43 [SYS.SRC]EXCEPTION.MAR;1   (26)

```
                      0252  1352  ;++
                      0252  1353  ; MCHK_TEST - TEST TO SEE IF RECOVERY BLCOK ENABLED AT TIME OF MACHINE CHECK
                      0252  1354  ;
                      0252  1355  ; INPUTS:
                      0252  1356  ;
                      0252  1357  ;     R1 - POINTER TO PC,PSL OF MACHINE CHECK EXCEPTION
                      0252  1358  ;
                      0252  1359  ; OUTPUTS:
                      0252  1360  ;
                      0252  1361  ;     R0 - .TRUE. IF RECOVERY BLOCK CURRENTLY IN EFFECT
                      0252  1362  ;          .FALSE. IF NO RECOVERY BLOCK
                      0252  1363  ;     ALL OTHER REGISTERS PRESERVED
                      0252  1364  ;
                      0252  1365  ;--
                      0252  1366
                      0252  1367  MCHK_TEST:
                      0252  1368
              50  D4  0252  1369          CLRL    R0                      ; ASSUME NO RECOVERY BLOCK
        0000'CF  D5  0254  1370          TSTL    W^MCHK$GL_SP            ; IF SP NOT ZERO, THEN RECOVERY BLCOK
              02  13  0258  1371          BEQL    40$
              50  D6  025A  1372          INCL    R0                      ; SET LOW BIT, INDICATING RECOVERY BLOCK
                      025C  1373  40$:
              05  025C  1374          RSB                             ; RETURN
```

```
                        025D   1376  ;++
                        025D   1377  ;
                        025D   1378  ; EXE$MCHK$_TEST - TEST RECOVERY BLOCK FOR MASK MATCH
                        025D   1379  ;
                        025D   1380  ; FUNCTIONAL DESCRIPTION:
                        025D   1381  ;
                        025D   1382  ;     TEST TO SEE IF MACHINE CHECK RECOVERY BLOCK EXEISTS WHEN MACHINE
                        025D   1383  ;     CHECK EXCEPTION OCCURRED.  IF SO, CHECK TO SEE IF FUNCTION MASK
                        025D   1384  ;     BITS MATCH.
                        025D   1385  ;
                        025D   1386  ; INPUTS:
                        025D   1387  ;
                        025D   1388  ;     R1 - POINTS TO PC,PSL PAIR ON STACK FROM MACHINE CHECK
                        025D   1389  ;     R2 - CONTAINS BITS TO TEST AGAINST MASK SPECIFIED BY PROTECTED CODE
                        025D   1390  ;
                        025D   1391  ; OUTPUTS:
                        025D   1392  ;
                        025D   1393  ;     R0 - .TRUE. IF RECOVERY BLOCK IN EFFECT AND MASKS MATCH
                        025D   1394  ;          .FALSE. IF NEITHER RECOVERY BLOCK OR MASK MATCH
                        025D   1395  ;     R2 - DESTROYED
                        025D   1396  ;     ALL OTHER REGISTERS PRESERVED
                        025D   1397  ;
                        025D   1398  ;--
                        025D   1399
                        025D   1400  EXE$MCHK_TEST::
                        025D   1401
              F3   10   025D   1402          BSBB    MCHK_TEST               ; RECOVERY BLOCK IN EFFECT HERE?
           0C 50   E9   025F   1403          BLBC    R0,50$                  ; NO, RETURN WITH R0 CLEAR
                        0262   1404
                        0262   1405          ASSUME  MCHK$M_LOG EQ 1
                        0262   1406
           0A 52   E8   0262   1407          BLBS    R2,60$                  ; HANDLE LOG BIT SEPARATELY
   0000'CF 52   D3   0265   1408  45$:       BITL    R2,W^MCHK$GL_MASK       ; MASKS MATCH?
              02   12   026A   1409          BNEQ    50$                     ; YES
              50   D4   026C   1410  47$:     CLRL    R0                      ; NO, R0 = .FALSE.
                 05   026E   1411  50$:       RSB                             ; RETURN
                        026F   1412
   F8 0000'CF   E9   026F   1413  60$:       BLBC    W^MCHK$GL_MASK,47$      ; ERROR LOG NOT DISABLED
              52   D7   0274   1414          DECL    R2                      ; WANT TO DISABLE LOGGING, CHECK REST
                        0276   1415                                           ; OF MASK FOR ERROR TYPE
              ED   11   0276   1416          BRB     45$
                        0278   1417
                        0278   1418          .END
```

```
ACVIOLAT                00000063 R    03          EXE$KERSTKNV              0000010C RG   03
ATTCONSTO_IDX         = 00000001                  EXE$MCHECK               00000110 RG   03
ATTCONSTO_MSG           00000033 R    02          EXE$MCHK_BUGCHK          0000021B RG   03
BADAST_IDX            = 00000003                  EXE$MCHK_PRTCT           000001F3 RG   03
BADAST_MSG              00000083 R    02          EXE$MCHK_TEST            0000025D RG   03
BADHANDLER_IDX       = 00000002                   EXE$OPCCUS               00000118 RG   03
BADHANDLER_MSG          00000052 R    02          EXE$OPCDEC               00000120 RG   03
BADSTACK                00000292 R    02          EXE$PAGRDERR             00000152 RG   03
BAD_AST                 00000287 R    02          EXE$PROBEW               ******** X    02
BAD_HANDLER             0000026D R    02          EXE$RADRMOD              0000015C RG   03
BUG$_FATALEXCPT         ********  X   02          EXE$REFLECT              000000A6 RG   02
BUG$_INVEXCEPTN         ********  X   03          EXE$ROPRAND              00000164 RG   03
BUG$_KRNLSTAKNV         ********  X   03          EXE$SIGTORET             000003F3 RG   02
BUG$_UNABLCREVA         ********  X   02          EXE$SRCHANDLER           00000201 RG   02
CHECK_FP                000003CC R    02          EXE$SSFAIL               00000180 RG   03
CHECK_STACK             00000328 R    02          EXE$TBIT                 0000016C RG   03
CHF$L_MCHARGLST      = 00000008                   EXE$UNWIND               ******** X    02
CHF$L_MCH_DEPTH      = 00000008                   EXSSXT                   00000187 R    03
CHF$L_MCH_FRAME      = 00000004                   FINALMSG                 00000010 R    02
CHF$L_MCH_SAVRO      = 0000000C                   FINAL_IDX              = 00000000
CHF$L_SIGARGLST      = 00000004                   HANDLER                = 00000000
CHF$L_SIG_NAME       = 00000004                   IPL$_ASTDEL            = 00000002
CONT_FROM_STOP          0000025C R    02          LIB$_ATTCONSTO           ******** X    02
COPYARGS                0000030E R    02          MCHK$GL_MASK             ******** X    03
CRESTACK                000001A2 R    02          MCHK$GL_SP               ******** X    03
CTL$AL_CMCNTX           ********  X   03          MCHK$M_LOG             = 00000001
CTL$AL_FINALEXC        ********  X   02           MCHK_TEST                00000252 R    03
CTL$AL_STACK           ********  X   02           MMG$PTEINDX              ******** X    02
CTL$AL_STACKLIM        ********  X   02           MSG_VECTOR               00000000 R    02
CTL$AQ_EXCVEC          ********  X   02           NORMAL                   000001EF R    02
CTL$GB_SSFILTER       ********  X   02            PCB$L_PHD              = 0000006C
CTL$GL_CMHANDLR       ********  X   03            PHD$L_FREP1VA         = 00000030
CTL$GL_CMSUPR         ********  X   03            PR$_IPL               = 00000012
CTL$GL_CMUSER         ********  X   03            PROTECT_RETURN           0000020E R    03
CTL$GL_PCB            ********  X   02             PSL$C_EXEC            = 00000001
CTL$GL_PHD            ********  X   02             PSL$C_SUPER           = 00000002
EX3ARG                  0000017B R    03          PSL$C_USER            = 00000003
EX4ARG                  00000106 R    03          PSL$M_CM              = 80000000
EX5ARG                  00000157 R    03          PSL$M_FPD             = 08000000
EXCCMD                  000000BF R    03          PSL$M_PRVMOD          = 00C00000
EXE$ACVIOLAT            00000000 RG   03           PSL$M_TBIT            = 00000010
EXE$ARITH               0000006C RG   03           PSL$M_TP              = 40000000
EXE$ASTDEL              ********  X   02           PSL$S_CURMOD          = 00000002
EXE$ASTFLT              00000077 RG   03           PSL$S_PRVMOD          = 00000002
EXE$BREAK               0000008C RG   03           PSL$V_CM              = 0000001F
EXE$BUG_CHECK           ********  X   03           PSL$V_CURMOD          = 00000018
EXE$CMODSUPR            00000094 RG   03           PSL$V_IS              = 0000001A
EXE$CMODUSER            000000AC RG   03           PSL$V_PRVMOD          = 00000016
EXE$COMPAT              000000CC RG   03           REFLECT                  0000019A R    03
EXE$CONTSIGNAL          0000024A RG   02           SAVAP                 = 00000008
EXE$EXCEPTION           0000018F RG   03           SAVFP                 = 0000000C
EXE$EXCMSG              ********  X   02            SAVMSK                = 00000006
EXE$EXPANDSTK           0000041A RG   02           SAVPC                 = 00000010
EXE$GL_ABSTIM           ********  X   03           SAVPSW                = 00000004
EXE$GL_BADACV_C         ********  X   03           SAVRG                 = 00000014
EXE$GL_BADACV_T         ********  X   03           SEARCH                   00000345 R    02
EXE$GL_VAXEXCVEC        ********  X   02           SS$_ACCVIO            = 0000000C
```

```
SS$_ARTRES        = 00000474
SS$_ASTFLT        = 0000040C
SS$_BADSTACK      = 000002B4
SS$_BREAK         = 00000414
SS$_CMODSUPR      = 0000041C
SS$_CMODUSER      = 00000424
SS$_COMPAT        = 0000042C
SS$_MCHECK        = 000002BC
SS$_NOHANDLER     = 000008F8
SS$_NORMAL        = 00000001
SS$_OPCCUS        = 00000434
SS$_OPCDEC        = 0000043C
SS$_PAGRDERR      = 00000444
SS$_RADRMOD       = 0000044C
SS$_RESIGNAL      = 00000918
SS$_ROPRAND       = 00000454
SS$_SSFAIL        = 0000045C
SS$_TBIT          = 00000464
SS$_UNWIND        = 00000920
SS$_WASSET        = 00000009
STS$K_SEVERE      = 00000004
STS$M_SEVERITY    = 00000007
STS$S_SEVERITY    = 00000003
STS$V_SEVERITY    = 00000000
SYS$ADJSTK          ********   X   02
SYS$CALL_HANDL      ********   X   02
SYS$CRETVA          ********   GX  02
SYS$DELPRC          ********   GX  02
SYS$DELTVA          ********   GX  02
SYS$EXIT            ********   GX  02
SYS$SETSFM          ********   GX  02
VA$V_P1           = 0000001E
VAFAIL              000001D5 R     02
```

```
+------------------+
! Psect synopsis !
+------------------+
```

PSECT name	Allocation		PSECT No.	Attributes
. ABS .	00000000 (0.)		00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)		01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
YEXEPAGED1	00000490 (1168.)		02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
$AEXENONPAGED	00000278 (632.)		03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

```
+----------------------------+
! Performance indicators !
+----------------------------+
```

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.10	00:00:00.75
Command processing	107	00:00:00.53	00:00:04.35
Pass 1	336	00:00:11.05	00:00:28.15
Symbol table sort	0	00:00:01.54	00:00:04.80
Pass 2	266	00:00:03.59	00:00:10.22
Symbol table output	18	00:00:00.13	00:00:00.13

B 10

EXCEPTION   - EXCEPTION HANDLING   16-SEP-1984 00:06:28 VAX/VMS Macro V04-00   Page 42
VAX-11 Macro Run Statistics   5-SEP-1984 03:41:43 [SYS.SRC]EXCEPTION.MAR;1   (27)

```
Psect synopsis output            1        00:00:00.02      00:00:00.02
Cross-reference output           0        00:00:00.00      00:00:00.00
Assembler run totals           759        00:00:16.97      00:00:48.74
```

The working set limit was 1800 pages.
66462 bytes (130 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 953 non-local and 66 local symbols.
1418 source lines were read in Pass 1, producing 22 object records in Pass 2.
28 pages of virtual memory were used to define 27 macros.

```
                                   +-------------------------------+
                                   ! Macro library statistics !
                                   +-------------------------------+


Macro library name                         Macros defined
----------------                           --------------
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                     9
_$255$DUA28:[SYSLIB]STARLET.MLB;2                 15
TOTALS (all libraries)                            24
```

1026 GETS were required to define 24 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:EXCEPTION/OBJ=OBJ$:EXCEPTION MSRC$:EXCEPTION/UPDATE=(ENH$:EXCEPTION)+EXECML$/LIB

FILEREAD
LIS

EXCEPTMSG
LIS

DISMOUNT
LIS

DEBUGDATA
LIS

ERRORLOG
LIS

DEVICEDAT
LIS

EXCEPTION
LIS

FILERWIO
LIS

EXSUBROUT
LIS